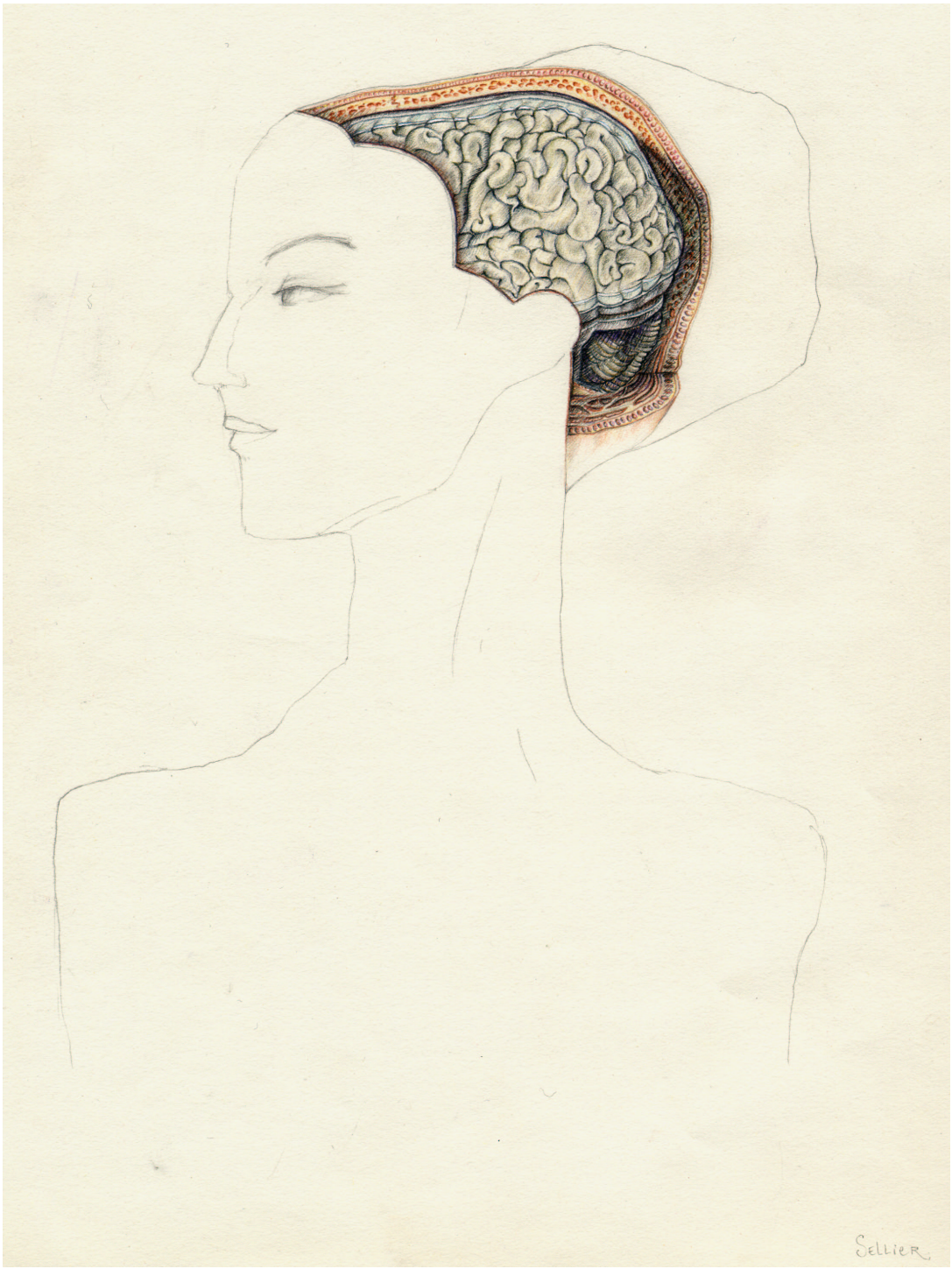
II

Extraction de Surfaces


Sellier

Chapter 4

Modèles Déformables pour l'Extraction de Surfaces en Imagerie Médicale

Résumé — Dans ce chapitre, nous revenons sur différentes méthodes d'extraction de surface en imagerie médicale. Partant du déjà connu modèle des *snakes* de *Kass, Witkin, et Terzopoulos* [82], en section 4.1, nous présentons des méthodes basées sur une représentation explicite de la surface, comme dans les travaux de *Delingette* [38]. Nous étudions ensuite en section 4.2 une représentation implicite de la surface, à partir des travaux de *Caselles, Kimmel, et Sapiro* [22]. Nous détaillons l'implémentation de ces contours actifs géodésiques dans la section 4.3, à partir de la méthode des Ensembles de Niveaux, développée et amplement détaillée dans le livre de *Sethian* [163]. On s'intéresse en section 4.4 à définir un modèle abstrait pour nos applications, et on poursuit par son implémentation à l'aide de modèle déformables explicites puis implicites. Nous présentons par la suite l'utilisation du *Fast-Marching* comme algorithme de segmentation dans la section 4.5, et nous détaillons les applications existantes de cette méthodologie à des problèmes classiques de segmentation en imagerie médicale 3D à la section 4.6.

Abstract — In this chapter, we recall the different methods used for surface extraction in medical imaging. Starting from the already studied *snakes* framework of *Kass, Witkin, and Terzopoulos* [82] in section 4.1, we mention methods based on explicit representations of the shape, as done by *Delingette* [38]. Thus we extend in section 4.2 to implicit representations of the shape, proposed by *Caselles, Kimmel, and Sapiro* [22]. We detail implementation of those Eulerian active contours in section 4.3, using the level-sets methodology developed extensively by *Sethian* [163]. We develop an abstract deformable model for our applications in section 4.4, and we follow by implementations of this framework with explicit and implicit deformable models. We further detail the use of the *Fast-Marching* method to segmentation tasks in section 4.5. And we detail several applications of this methodology to medical image segmentation problems in section 4.6.

4.1 Classical Active Contours

4.1.1 Definition

We recall the *Snake* model as introduced in [82], and already mentioned in chapter 1. For a general overview on deformable models, see [115]. See also a description and references in [30].

The classical energy of the model which will be minimized on $\mathcal{A}$ the space of all admissible curves, and has the following form:

$$\begin{aligned} E: \quad & \mathcal{A} \to \mathbb{R} \\ & \mathcal{C} \mapsto E(\mathcal{C}) \quad = \int_\Omega \frac{w_1}{2}\|\mathcal{C}'(v)\|^2 + \frac{w_2}{2}\|\mathcal{C}''(v)\|^2 + P(\mathcal{C}(v))dv \end{aligned}$$

where $\Omega = [0,1]$ is the parameterization interval. This model is used in the classical formulations of deformable models [32, 102].

In this formulation, each term appears as a potential acting on the shape. Thus the mechanical properties of the deformable model are controlled by two kinds of constraints:

- The internal potential: C' and C'' are the smoothing terms on the curve. They enable to control its regularity by means of w_1 which quantifies its rigidity and w_2 its elasticity;
- The external potential term P represents the likelihood. This "image" potential traps the curve towards the regions with desired attributes.

Figure 4.1[1] displays iterations of the heart segmentation in a 3D ultrasound image of the heart, with simplex meshes.

4.1.2 Drawbacks

The main drawbacks of the classical deformable model approach are:

- Minimization: The functional is non-convex, and one difficulty is to find a good local minimum. Spurious edges generated by noise may stop the evolution of the surface, giving an insignificant local minimum of the energy;
- Initialization: The user must specify an initial shape that is close to the goal, like a very precise polygon approximation, which may be tedious to draw;
- Topology changes: this method is unable to segment several objects simultaneously, and to merge different shapes;

One of the main issue in using deformable models is their initialization and minimization. The solution introduced in [163] allows to solve the global minimization of a problem. His approach considers a slightly modified problem: a curve is considered as an interface between two media, following a particular evolution equation. We will see that under precise assumptions, this front evolution efficiently builds a path between two fixed points, as detailed in [34].

[1] Slice of a 3D ultrasound dataset acquired with a multi-plane trans-oesophagus scan-head on a HDI 5000 ATL echograph.

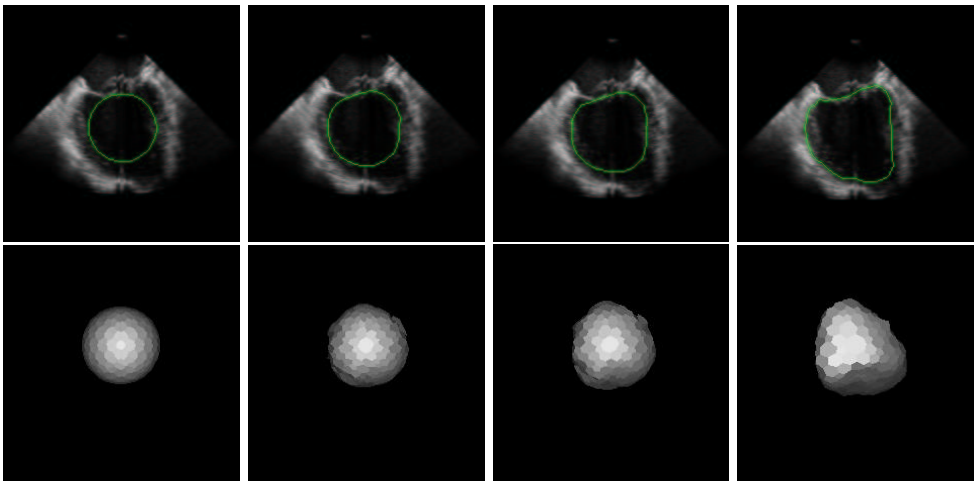


Figure 4.1. Samples of the segmentation of a heart in a 3D ultrasound image with simplex meshes [37] : Starting with a sphere interpolated with simplex mesh faces, we segment a heart in a 3D ultrasound image: first row shows the intersection of the mesh with a slice of the dataset, and second row is the 3D model at the same iterations.

4.2 Geodesic Active Contours

A geometric approach for deformable models was introduced by *Caselles, Catté and Dibos* [21] and *Malladi, Sethian and Vemuri* [113]. The basic idea of the geometric model is that the curve follows an evolution by expansion in the normal direction, with lower speed when the image force $P(\mathcal{C})$ is small. Hence the evolution of a planar curve C is in the direction of its normal only is given by

$$\frac{\partial \mathcal{C}(s,\tau)}{\partial \tau} = P(\mathcal{C})(\frac{\partial^2 \mathcal{C}}{\partial s^2} + w\ \mathbf{n}) = P(\mathcal{C})(\kappa + w)\mathbf{n}, \tag{4.1}$$

where s is the arc-length parameter of the curve $\mathcal{C}$, κ is the curvature, $\mathbf{n}$ is the unit normal. The constant term w is similar to the balloon force introduced in the *snakes* model [29] (and also related to the dilatation transform in mathematical morphology and the grass-fire transform [102]).

It was shown that the geometric snakes model handles topology changes better than the classical snakes when implemented with the level set approach for curve evolution proposed by *Osher and Sethian* [135, 158].

But, due to the formulation of the image force, it never comes to a complete stop, and heuristic stopping procedures are used to switch off the evolution process when an edge is reached. In equation (4.1), the geometric snake evolution is slower when the P is small but the curve does not necessary stop completely at the boundary, since it never reaches an equilibrium.

Given an initial curve $\mathcal{C}(s,0)$, the *geodesic active contours* is based on the planar

evolution equation

$$\frac{\partial \mathcal{C}(s,\tau)}{\partial \tau} = (P(\mathcal{C})(\kappa + w) - \langle \nabla P, \mathbf{n} \rangle)\, \mathbf{n}, \tag{4.2}$$

where s is the arclength. The ∇P term added, in comparison to the geometric model, is a projection of the attraction force $-\nabla P$ on the normal to the curve. This force balances the other term close to the boundary and causes the curve to stop there.

This introduction of ∇P, based on geometrical as well as energy minimization reasoning, leads to the "geodesic active contour" proposed by *Caselles, Kimmel and Sapiro* [23]. The geodesic active contours enjoy the advantages of classical as well as geometric active contours, since it handles topology changes and reaches an equilibrium which is similar to the classical snakes.

The curve evolution equation is then reformulated and implemented using the *Osher-Sethian* numerical algorithm [135]. Similar geometric models were also introduced in [85, 183, 165, 149] and extended to color and texture in [153].

4.3 *Level-Sets* Implementation of the Geodesic Active Contours

Let $C_o(p)$ be a closed initial parameterized planar curve in an Euclidean plane, and $C(p,t)$ the family of curves generated by the movement of $C_0(p)$ in the direction of its outward Euclidean normal vector $\mathbf{n}$. The speed of this movement is supposed to be a scalar function of the curvature κ:

$$\begin{cases} \frac{\partial C}{\partial t}(p) &= F(\kappa)\mathbf{n} \\ C(p,0) &= C_0(p) \end{cases} \tag{4.3}$$

A Lagrangian approach might be considered to implement the curve evolution according to the above equations in motion equations of the discretized positions of $C(p)$.

But this formulation has several drawbacks. When tracking the motion of the interface C propagating along its normal direction with velocity F, most numerical techniques rely on markers, breaking it up into points connected by segments, and moving each point with speed F (see figure 4.2-left). The solution is supposed to gain in accuracy if the number of points is increased. Problems arise if different parts of the front cross each other, or if the shape tries to break into two pieces (see figure 4.2-middle and the book of *Sethian* [163] for details) or if two shapes try to merge into one (see figure 4.2-right).

Therefore, the main drawback of this approach is that the evolving model is not capable to deal with topological changes of the moving front, and external procedures must be added to detect and deal with merging and splittings. These methods were developed for active contour models, namely *Topological Snakes* (T-snakes) in [114], for triangulated meshes in [97], and for their dual simplex meshes in [121]. There is no suitable difference approximation scheme for the Lagrangian implementation, due to the time-dependent parameterization of the curve model, thus leading to stability problems.

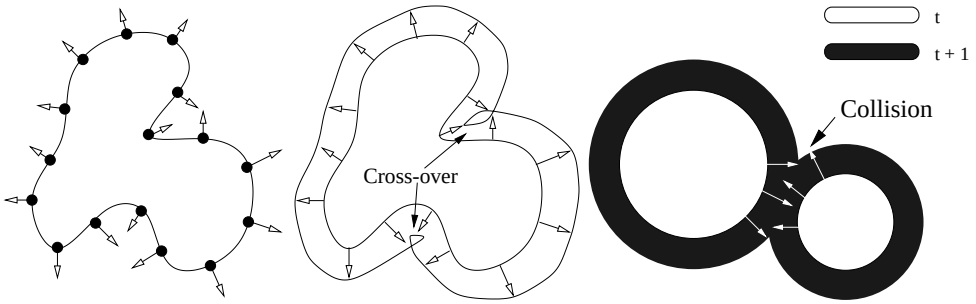


Figure 4.2. Markers methods and related problems: Left image illustrates the markers technique. Middle image shows the cross over which occur frequently. Right image shows problems of merging two contours.

These problems are handled very elegantly by the level set methodology, originally introduced by *Osher and Sethian* in [135], and now widely used in lots of applications, ranging from computer vision problems, to semiconductor manufacturing, shape from shading, and robotic navigation (see [163]). They embed the initial position of the moving interface $C_0(\mathbf{x})$ as the zero level set of a higher dimensional function ϕ (the signed distance to C_0, as shown in figure 4.3), and link the evolution of this new function ϕ to the evolution of the interface itself through a time-dependent initial value problem. At each time t, the contour $C(t)$ is given by the zero level-set of ϕ.

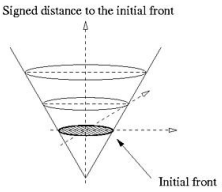


Figure 4.3. Embedding the contour in the signed distance: In this 2D example, the initial contour $C_0(x, y)$ is a circle, and $\phi(x, y, 0) = \pm d(C_0((x, y))$.

This condition states that

$$\phi(C(t), t) = 0 \Rightarrow \phi_t + \nabla\phi(C(t), t) \cdot \frac{\partial C}{\partial t} = 0 \tag{4.4}$$

Since $\frac{\partial C}{\partial t} = F\mathbf{n}$ and the outward normal vector is given by $\frac{\nabla\phi}{|\nabla\phi|}$, this yields the following evolution equation for ϕ given in [135]:

$$\begin{cases} \phi_t + F|\nabla\phi| = 0 \\ \phi(\mathbf{x}, 0) = C_0(\mathbf{x}) \end{cases} \tag{4.5}$$

4.3.1 Advantages of this formulation

There are several advantages associated with the *Level-Sets* paradigm:

1. This formulation remains unchanged in higher dimensions, as well for 2D curves, as for hypersurface in three dimensions (and higher). Therefore the embedded hypersurface will be denoted Γ in the following.

2. The evolving function ϕ remains a functions as long as F is smooth. As a consequence, topological changes in the evolving front $\Gamma(\mathbf{x}, t)$ are handled naturally, the position of the front at time t is given by the zero level-set $\phi(\mathbf{x}, t) = 0$ of the evolving function ϕ. $\Gamma(\mathbf{x}, t)$ can be several initial curves and it can break and merge as t advances. A 2D examples of fronts merging is shown in figure 4.4.

Figure 4.4. Easy handling of contour merging with *Level-Sets* : $\Gamma(x, y, t = 0)$ is the representation of the three curves. Initializing ϕ by the distance to these circles, and propagating ϕ with a constant positive speed in the outward normal direction, $\phi_t + \beta|\nabla\phi| = 0$, the three circles increase and merge. The different images represent the zero level-set of ϕ at several successive iterations.

3. Intrinsic geometry properties of the front are easily determined from the front itself, like the normal to the front $\frac{\nabla\phi}{|\nabla\phi|}$, and the curvature of the front $\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|}$.

4. The evolution equation (4.5) can be approximated by efficient computational schemes with finite different approximation for the spatial and temporal derivatives. Explicit finite difference approach is possible, and other implicit and semi-implicit approaches have been already developed [65]

Explicit representation of the surfaces can also handle topology changes. Several attempt to achieve this tasks were developed independently.

- *McInerney and Terzopoulos* [114, 116] propose the *"T-snakes"* and *"T-surfaces"* with adaptive topology; the initial model is a triangulation that evolved according to a Lagrangian evolution equation, and the triangulation is resampled by computing its intersection with a tetrahedral griding of the image domain. By defining an "inside" and an "outside" region, the resampling handles topology changes when the surface self-intersect; this ad-hoc approach is limited to closed contours and surfaces.

- *Lachaud et al.* [98, 97] propose a very interesting technique based on the distance between each vertices of the triangulation. But knowing this distance between each pair of nodes is a huge computing task;

- *Montagnat and Delingette* [39, 40, 121] propose topological changes based on the simplex mesh surface representation [37]. This method that resamples the surface on a regular grid of the image domain works well in 2D and works with low computation times, but no 3D extension is available for the moment.

The different methods mentioned are all proposing methods to adapt the topology of their objects. This method has advantage of reducing the importance of the *a priori* on the final shape of the target of the segmentation process. Therefore, the topology of the model at initialization do not need to be the same than the model at convergence, thus reducing user interaction. But these methods are based on approximations, while the *Level-Sets* formalism handles naturally this problem. The *Level-Sets* model is evolved, and its zero level-set can be represented at several iterations, as shown in figure 4.4. But this implicit representation has one major drawback: it severely limits the possible interactivity of the user on the model, as mentioned in [120]. However, we will see in the next chapter that some basic interactions can be applied to this formalism.

4.3.2 Different Motions of the interface

The evolution of the segmentation is performed through the evolution of ϕ, which is done by a flow equation:

$$\frac{\partial \phi}{\partial t} + F|\nabla \phi| = \frac{\partial \phi}{\partial t} + \mathbf{V} \cdot \nabla \phi = 0 \tag{4.6}$$

if we consider the vector flow field $\mathbf{V}(\mathbf{x}, t)$ with $(\mathbf{x}, t) \in \Omega \times [0, +\infty[$, which evolves $\phi(\cdot, t)$.

Scalar flow

Any flow of the form:

$$\mathbf{V} = \beta(\mathbf{x}, t)\mathbf{n} \text{ with } \beta(\mathbf{x}, t) \in \mathbb{R} \tag{4.7}$$

will be referred as a scalar flow. Evolution under (4.6) with positive (resp. negative) values for β yields to a normal dilatation (resp. shrinkage) of $\phi(0, t)$. Figure 4.8 displays iterations of a front, initialized with the distance to a circle, evolving under an inflating term. In figure 4.5, ϕ is evolved with $\beta = -1$ in the expression of the scalar vector field of equation (4.7). Scalar flows lead to non-linear flow equations.

Vector Flow

Any flow of the form:

$$\mathbf{V} = \mathbf{U}(\mathbf{x}, t) \text{ with } \mathbf{U}(\mathbf{x}, t) \in \mathbb{R}^{\mathrm{d}} \tag{4.8}$$

and no dependency on ϕ will be referred to as a vector flow, or vector flow field. Evolution under (4.6) corresponds to passive advection of the level sets of $\phi(\cdot, t)$. Figure 4.6 displays iterations of a front, initialized with the distance to a circle, evolving under the influence of an advection flow only with $U(\mathbf{x}, y, t) = \mathbf{u} = (1, 1)$. The resulting contour is implicitly moved, in the direction of the vector $\mathbf{u}$. Vector flows lead to linear flow equations.

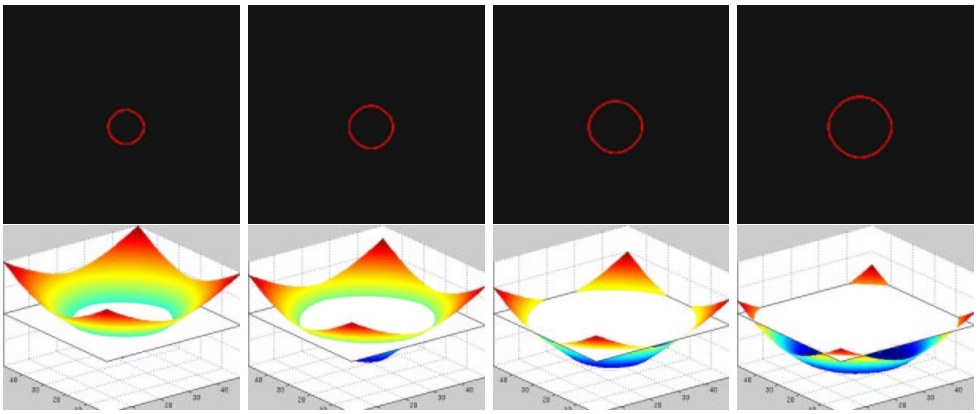


Figure 4.5. Inflating force: First row images are consecutive iterations of the *Level-Sets* evolution under external forces only; The external forces are built using the gradient of a left-ventricle image; Bottom row shows the zero level-set superimposed on the gradient magnitude at the same iterations.

Curvature motion

Any flow of the form:

$$\mathbf{V} = -\varepsilon(\mathbf{x}, t)\, \kappa_M(\mathbf{x}, t)\, \mathbf{n} \text{ with } \varepsilon(\mathbf{x}, t) \in \mathbb{R} \tag{4.9}$$

will be referred to as a curvature flow. Evolution under equation (4.6) with positive values for ε yields to a local regularization of $\phi(0, t)$. Negative values for ε lead to instabilities. Figure 4.7 displays iterations of a front, initialized with the signed distance to an initial curve, evolving under the influence of its curvature only, using the level set equation with a speed function of the form $F(\kappa) = -\kappa$

$$\phi_t = \kappa |\nabla \phi| = \left[\nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \right] |\nabla \phi| \tag{4.10}$$

If we let the iterations proceed, the zero level-set (in black) will tend to a circle, and shrink to a point before vanishing. Equation (4.10) applied to this image illustrates Grayson's theorem that all simple closed curves moving under its curvature must shrink to a point (as shown in [70]), regardless of its initial shape. This motion resembles a non-linear heat equation (see [163]) and smoothes large oscillations, and can be used in curve extraction as a diffusion term, to relax boundaries. Curvature flows lead to non-linear flow equations.

Composite flow

Any flow which is the sum of flows of the preceding types will be referred to as a composite flow. Figure 4.8 displays iterations of a front, initialized with the distance

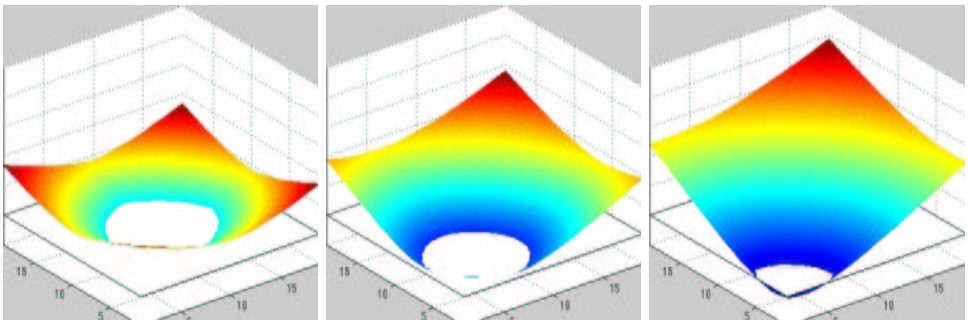


Figure 4.6. Advection flow: the level sets are represented using a colored ladder; the zero level-set is implicitly defined by the intersection between the levels and the white plan at height zero.

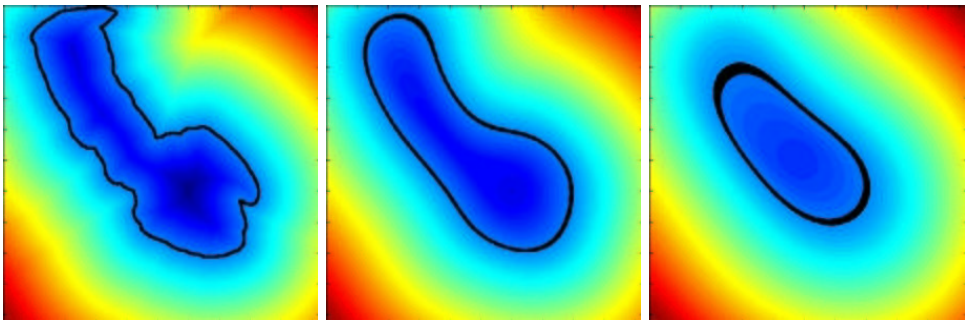


Figure 4.7. Curvature motion: the zero level-set (in black) and all the other levels evolve according to their curvature.

to a circle, evolving under the following equation:

$$\phi_t + g_I(1 - \varepsilon\kappa)|\nabla\phi| - \beta\nabla P \cdot \nabla\phi \tag{4.11}$$

where the equation contains the following terms:

1. an inflating force, as in equation (4.7), here defined by

$$g_I(\mathbf{x}) = \frac{1}{1 + |\nabla I_\sigma(\mathbf{x})|} \tag{4.12}$$

   where I_σ is the image convolved with a Gaussian kernel of size σ; this inflating force vanishes to zero in region of high gradients (i.e. near object boundaries)

2. a curvature term $-g_I\varepsilon\kappa$ which controls the smoothness of the iso-contours of $\phi(\cdot, t)$, originally introduced in [112];

3. an external force, which role is to attract the surface towards the boundary of the object of interest. This term is a vector flow which denotes a projection of an attractive force vector on the surface normal, in our case we use a potential field defined as in [22], by

$$P(\mathbf{x}) = -|\nabla I_\sigma(\mathbf{x})| \tag{4.13}$$

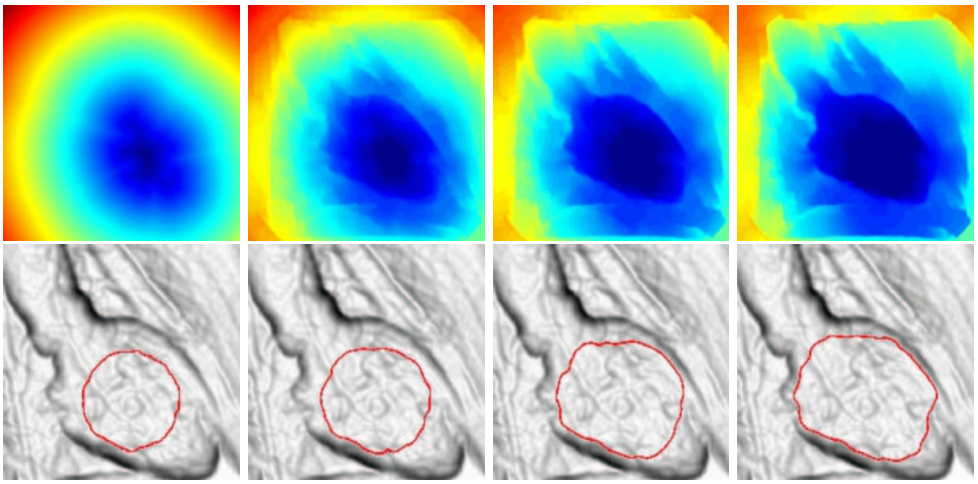


Figure 4.8. Composite flow: First row images are consecutive iterations of the *Level-Sets* evolution under the equation (4.11); the external forces are built using the gradient of a left-ventricle image; bottom row shows the zero level-set superimposed on the gradient magnitude at the same iterations.

4.4 Region-based forces

In this chapter we present a variational framework for the segmentation on the basis of the geodesic contour implementation of region-based forces. This framework has been used in the following parts of the thesis.

4.4.1 Partitioning the image domain

Region-based terms have already been included in active contour models [148, 27, 25, 24] We consider an *abstract deformable model*, which consists of a partition of the *image domain* Ω into three subsets: two open subsets $\Omega_{in}(t)$ and $\Omega_{out}(t)$ (the *inside* and the *outside* of the segmented object) and their common boundary $\Gamma(t)$. $\Gamma(t)$ is supposed to be as smooth as necessary (for example we may suppose that $\Gamma(t)$ is a locally Lipschitz-continuous hypersurface). This partition is a function of an evolution

parameter $t \in [0, +\infty[$ that will be called *time*[2]:

$$\begin{cases} \Omega = \Omega_{in}(t) \cup \Gamma(t) \cup \Omega_{out}(t) \\ \Gamma(t) = \partial\Omega_{in}(t) = \partial\Omega_{out}(t) \end{cases} \tag{4.14}$$

Figure 4.9 shows an example of 2D image, with the abstract deformable model as we would like it to look like after segmenting the image.

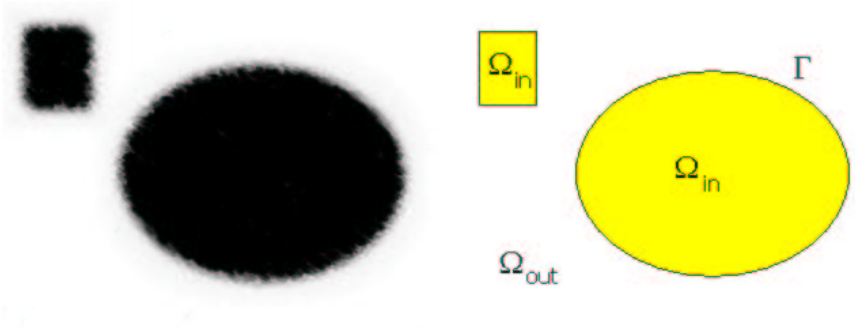


Figure 4.9. Abstract Deformable Model: left image is the 2D synthetic data; right image is the 2D abstract deformable model.

4.4.2 Region descriptors

Region descriptors are real functions of the image intensity I, and of Ω_{in} and Ω_{out}. We consider:

$$\begin{cases} k_{in}(\mathbf{x}, t) = -\log P_{in}(\mathbf{x}, t) \\ k_{out}(\mathbf{x}, t) = -\log P_{out}(\mathbf{x}, t) \end{cases} \tag{4.15}$$

where $P_{in}(\mathbf{x}, t)$ (resp. $P_{out}(\mathbf{x}, t)$) is the probability that x is in $\Omega_{in}(t)$ (resp. $\Omega_{out}(t)$).

Gaussian descriptors were first introduced by *Zhu and Yuille* [196]. They are "fully automatic" in the sense that no user-defined parameter is necessary to their definition. They are based on region probabilities defined by Gaussian distributions:

$$P_{in}(\mathbf{x}, t) = \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)}\ \exp(-\frac{(I(\mathbf{x}) - \mu_{in}(t))^2}{2\,\sigma_{in}^2(t)}) \tag{4.16}$$

A similar expression is used for the definition of $P_{out}(\mathbf{x}, t)$. Here $\mu_{in}(t)$ and $\sigma_{in}^2(t)$ are respectively the mean and the variance of the image intensity over $\Omega_{in}(t)$:

$$\mu_{in}(t) = \frac{\int_{\Omega_{in}(t)} I(\mathbf{x})\,d\mathbf{x}}{\int_{\Omega_{in}(t)}\ d\mathbf{x}}\ ,\ \sigma_{in}^2(t) = \frac{\int_{\Omega_{in}(t)} (I(\mathbf{x}) - \mu_{in}(t))^2\,d\mathbf{x}}{\int_{\Omega_{in}(t)}\ d\mathbf{x}}$$

This means that the histograms of the intensity in $\Omega_{in}(t)$ and $\Omega_{out}(t)$ are *modeled* by Gaussian distributions. Those descriptors were also used extensively when *Paragios*

[2]this "artificial" time has nothing to do with the notion of physical time.

introduced the concept of *Geodesic Active Regions* [137], for unsupervised image segmentation, supervised texture segmentation [139], and detection and tracking of moving objects [138].

4.4.3 Boundary descriptors

The boundary descriptor k_b, which is sometimes called *edge indicator*, is usually a real function of ∇I. We took the classical expression inherited from geodesic active contours (see for example *Caselles et al.* [23] or *Paragios* [137]):

$$k_b(\mathbf{x}) = \frac{1}{1 + \frac{\|\nabla I(\mathbf{x})\|^2}{\gamma^2}}$$

where γ is a user-defined parameter.

To make this descriptor "fully automatic", it is possible to use the mean gradient magnitude in the entire image and set:

$$\gamma = \frac{\int_{\Omega(t)} \|\nabla I(x)\| \, d\mathbf{x}}{\int_{\Omega(t)} d\mathbf{x}}.$$

This choice gave very good results on almost all the images we segmented.

Among other possibilities for boundary-based descriptors, we must mention the excellent work of *Xu and Prince* [189, 191, 190]. The construction of a new gradient information, using a somehow anisotropic diffusion technique to extend the local gradient information in the whole image, has been recently used together with the *Level-Sets* formalism in [140].

4.4.4 Segmentation as an optimization process

Composite energy functional

We see the process of segmentation of the image as the minimization of an *energy functional*, also called *objective function*. We decided to use a composite energy functional which comprises *region-based* and *boundary-based* terms. We took:

$$J(t) = \zeta \int_{\Omega_{in}(t)} k_{in}(\mathbf{x}, t) \, d\mathbf{x} + \zeta \int_{\Omega_{out}(t)} k_{out}(\mathbf{x}, t) \, d\mathbf{x} + \eta \int_{\Gamma(t)} k_b(\mathbf{x}) \, d\sigma \qquad (4.17)$$

where $d\mathbf{x}$, $d\sigma$, ζ and η are respectively the Lebesgue measures of $\mathbb{R}^{\mathrm{d}}$ and $\Gamma(t)$, and two user-defined positive scalar parameters.

The integrands in 4.17 are called *descriptors*, and contain all of the information that is being extracted from the image. The functions k_{in} and k_{out} are the *region descriptors*, and k_b is the *boundary descriptor*. The punctual values of the region descriptor k_{in} (resp. k_{out}) are supposed to be all the smaller as the probability to be in the interior (resp. exterior) of the object to be segmented is high. Similarly, the punctual values of the boundary descriptor are supposed to be all the smaller as the probability to be near a physical contour in the image is high.

The scalar parameters ζ and η may be adjusted to give more weight to the region-based integrals or the boundary-based integrals. The choice of ζ and η depends on the application specificities (quality and contrast of the images, image scale[3], etc.) and of the confidence of the user in the different descriptors.

Minimization of the energy functional

The evolution of the partition is totally defined by the evolution of the interface Γ between $\Omega_{in}(t)$ and $\Omega_{out}(t)$. The evolution (or motion) of Γ is defined by a speed field $\mathbf{V}(p,t)$, where p is the parameter corresponding to the chosen parameterization of Γ:

$$\frac{\partial \Gamma(p,t)}{\partial t} = \mathbf{V}(p,t). \tag{4.18}$$

We recall that the evolution of a hypersurface under an intrinsic (i.e. independent of the parameterization of the hypersurface) speed field depends only on the normal component of the speed (see Keriven [84, p. 40]).

The minimization of J is done by a gradient descent defined by the evolution equation (4.18). Several proofs have been proposed for the derivation of J. The most correct approach can be found in [79]. It shows that the evolution equation (4.18) implies that the temporal derivative of J is given by:

$$\begin{aligned}\frac{dJ}{dt} = \quad & \zeta \int_{\Omega_{in}(t)} \frac{\partial k_{in}}{\partial t}\, d\mathbf{x} + \zeta \int_{\Omega_{out}(t)} \frac{\partial k_{out}}{\partial t}\, d\mathbf{x} + \\ & \int_{\Gamma(t)} (\zeta\,(k_{in} - k_{out}) + \eta\,(k_b\,\kappa_M(p,t) + \nabla k_b))(\mathbf{V}.\,\mathbf{n})\, d\sigma\end{aligned}$$

where $\mathbf{n}$ and κ_M are respectively the normal (directed from $\Omega_{in}(t)$ to $\Omega_{in}(t)$) and the mean curvature of the hypersurface $\Gamma(t)$. In the case of time-independent region descriptors, the speed that minimizes the energy functional is consequently given by:

$$\mathbf{V} = \zeta\,(k_{out} - k_{in})\,\mathbf{n} - \eta\,(k_b\,\kappa_M(\mathbf{x},t)\,\mathbf{n} + \nabla k_b). \tag{4.19}$$

The evolution equation (4.18) will be embedded into the *Level-Sets* formulation described in section 4.3, and the flow (4.19) will be implemented using the different flows detailed in section 4.3.2. Note that $\mathbf{V}$ is intrinsic since it depends only on intrinsic features of $\Gamma(t)$. In the case of time-dependent region descriptors, other additive terms appear in (4.19) because of the two first terms in $\frac{dJ}{dt}$. But in the particular case of the Gaussian descriptors defined by (4.15) and (4.16), it is relatively easy to prove[4] that the two first terms in $\frac{dJ}{dt}$ are null, and that the speed that minimizes J is still given by (4.19).

The segmentation process consists in applying the evolution equation defined by (4.18) and (4.19) to a user-defined initial condition:

$$(\Omega_{in}(0), \Omega_{out}(0), \Gamma(0)) = (\Omega^0_{in}, \Omega^0_{out}, \Gamma^0)$$

The result of the segmentation process is given by the "limit" of the triplet $(\Omega_{in}, \Omega_{out}, \Gamma)$.

[3] Gaussian pre-smoothing of the image.

[4] All the vector analysis theorems needed for the proof are recalled in [79].

4.5 Segmentation with *Fast-Marching* algorithm

If we consider in equation (4.5) the particular case of a motion equation with a speed function $F > 0$, hence the interface always moves outward. One way to characterize the position of the interface is to compute the arrival time $T(\mathbf{x})$ of the interface as it crosses each point. Embedding this hypothesis in the level-set framework, a new equation is derived that determines the evolution of the surface $T(\mathbf{x})$ given by

$$\begin{aligned} T(C(\mathbf{x},t)) = t \quad &\Rightarrow \quad \nabla T \cdot C_t = 1 \\ &\Rightarrow \quad \nabla T \cdot \left(F \frac{\nabla T}{|\nabla T|} \right) = 1 \\ &\Rightarrow \quad F \cdot |\nabla T| = 1 \end{aligned} \tag{4.20}$$

that can be interpreted as: the gradient of arrival time is inversely proportional to the speed of the interface. This equation is the stationary case of the Hamilton-Jacobi formulation for propagating fronts where the time is disappeared, and if the speed is a function of the position of the front only, the equation reduces to what is known as the *Eikonal equation* equation, extensively studied in part I.

This particular equation (4.20) has also been used for surface extraction, since it has the same advantages as the *Level-Sets* formulation, detailed in section 4.3. In [111], *Malladi and Sethian* use the *Fast-Marching* algorithm in order to give a fast and rough initialization to a costly segmentation *Level-Sets* formulation. The *Level-Sets* model is based upon a composite flow, as shown in equation (4.11), for the segmentation of the cortex. The result of this segmentation was popularized by being advertised on the front cover of the *American Scientist Journal* [162]. They use as a speed function the following expression

$$F(\mathbf{x}) = e^{-\alpha |\nabla I_\sigma(\mathbf{x})|}, \; \alpha > 0 \tag{4.21}$$

where the speed has values very close to zero near high image gradients of the smoothed image I_σ. This expression is input in the *Eikonal equation* equation

$$|\nabla T(\mathbf{x})| = \frac{1}{F(\mathbf{x})} \tag{4.22}$$

The *Fast-Marching* in three dimension (see section 2.1) is then employed to march ahead. This helps to construct very quickly a good initial guess on the final surface. Then they input the final function $T(\mathbf{x})$ as an initial condition $\phi(\mathbf{x}, t = 0) = T(\mathbf{x})$, and iterate equation (4.11) a few time-steps with finite difference approximation schemes.

In practice the marching method is employed to march until a fixed time or until the size of the heap does not change very much (see section 1.3.2 for details on the heap used in the *Fast-Marching* algorithm) between two successive time increments.

4.6 Medical imaging applications of the *Level-Sets*

The *Level-Sets* models have already been used for a wide variety of applications, among them stereo problem [50], image classification [152], tracking of moving objects

[78], or modeling deformations of solid objects [185] among others. But the most important set of applications of the *Level-Sets* has been done in medical imaging, where their interesting properties in handling topology changes are very useful for segmenting the very complex anatomical shapes.

4.6.1 Cortex segmentation

Since initial contributions on 3D segmentation of medical images using the *Level-Sets* models by *Malladi and Sethian* [110, 111], the very complex shape of the cortical surface has attracted numerous contributions, due to the interesting properties of the *Level-Sets* formulation. One example of brain tissue segmentation is shown in figure 4.10, where the algorithm used is the *Fast-Marching* on the basis of the work presented in [111].

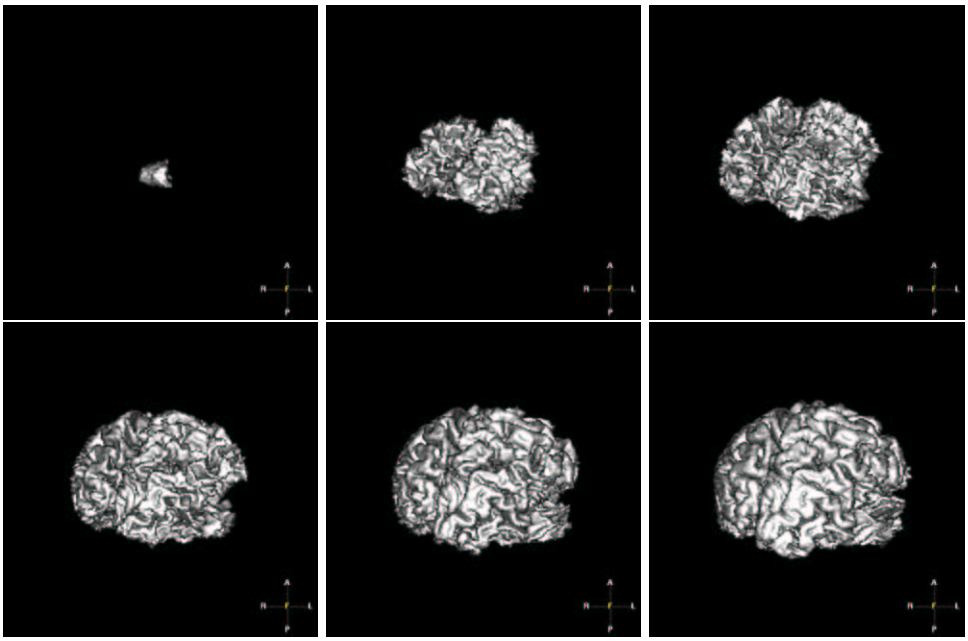


Figure 4.10. Segmentation result on the brain: On the basis of the model defined in [111], we segment the brain complex surface with the *Fast-Marching* algorithm, using a speed which is a simple distance function to an *a priori* mean value inside the white matter of the brain; formally $\mathcal{P}(\mathbf{x}) = \max\left(I(\mathbf{x}) - I_{mean}, 0\right) + w$.

In [67], authors proposed a fast implementation of the *Level-Sets* based on a semi-implicit formulation of the discretized evolution equation, based on similar use of these schemes for anisotropic diffusion filtering in [182].

An original method is presented in [7] where the authors propose to segment brain surfaces using sequentially a registration technique, and a *Level-Sets* model. However, it is not a real cooperation between both techniques, since the initialization

of any parameter of the *Level-Sets* are tuned on the basis of the first results obtained through the registration process. We will see in the following that the *Level-Sets* can cooperate with other fast segmentation techniques to enhance their preliminary results.

Several works have been developed on the basis of the simultaneous segmentation of the grey and white matter of the brain, using *coupled curve evolution equations* for segmenting both complex surfaces [193]. Main idea is to introduce a term relative to the distance between the two surfaces [194, 195]. In [68], this method is improved with a new scheme that computes the evolution of a new function which remains a distance function across iterations, and leads to better measurements of the distance between the two surfaces. And in [66] authors introduced the first geometric variational formulation for the coupled surfaces.

4.6.2 Brain Vessels

In the same region of the body, brain vessels extraction has attracted lots of attention from the computer vision community, with developments of very interesting dedicated *Level-Sets* implementations for thin tubular structure extraction.

We made a test on brain vessel extraction: the model is based on Gaussian region descriptors, and the initialization for k_{in} and k_{out} has been done using a **MDL** (minimum description length), with hypothesis that the image grey level information is a mixture of two Gaussian distributions. Descriptors are shown in figure 4.12. In figure 4.12 we show iterations of the segmentation of the brain vessels, in the

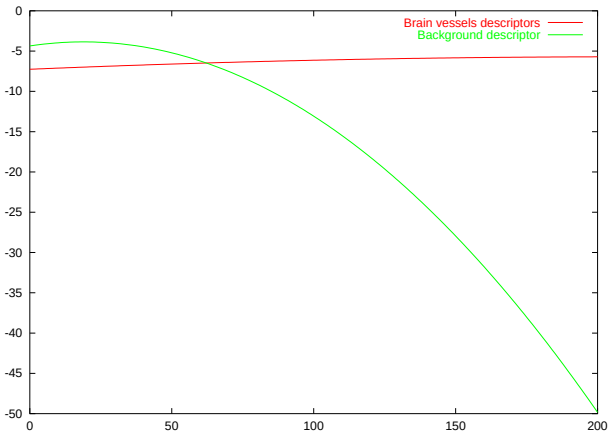


Figure 4.11. Gaussian region descriptors for brain vessels: assuming that the image is a mixture of two Gaussian distribution, the **MDL** classical computation of mean and variances has led to a model with ($\mu_{in} = 211.09, \sigma_{in} = 119.902$) and ($\mu_{out} = 18.9269, \sigma_{out} = 18.8822$).

dataset already displayed in figure 3.17, with the descriptors shown in figure 4.12. The propagation starts from one part of the vessels and propagates in the whole set.

Computing time for extracting the complete brain vessels is more than an hour, an a standard Sun workstation (300 MHz cpu).

In this field, *Lorigo et al.* [107] have developed flows using the Hessian information [108], co-dimension 2 evolution scheme for the extraction of thin curves in 3D [106], with application to the brain vessels. Same target was followed in [177] where the authors use a flow based on the divergence of the gradient in the image. Both works achieve extraction of thin curves where even *Level-Sets* classical formulation, as shown in figure 4.12, fails. However, the computing cost to achieve is too important. In the next chapter, we will settle another framework for segmentation in interactive computing time.

4.7 Summary

In this chapter we have explained an abstract deformable representation of the boundary between two regions, as used in [196]. The formulation of the evolution of the interface between the regions, defined with region descriptors, has been already studied for image segmentation in [137] with *Level-Sets* models. In the following we are going to develop improvements of this framework, introducing user interactivity on the interface, developments of algorithms to accelerate the speed of the computations, using the *Fast-Marching* first used for segmentation in [111]. On this basis we will show several applications of our framework.

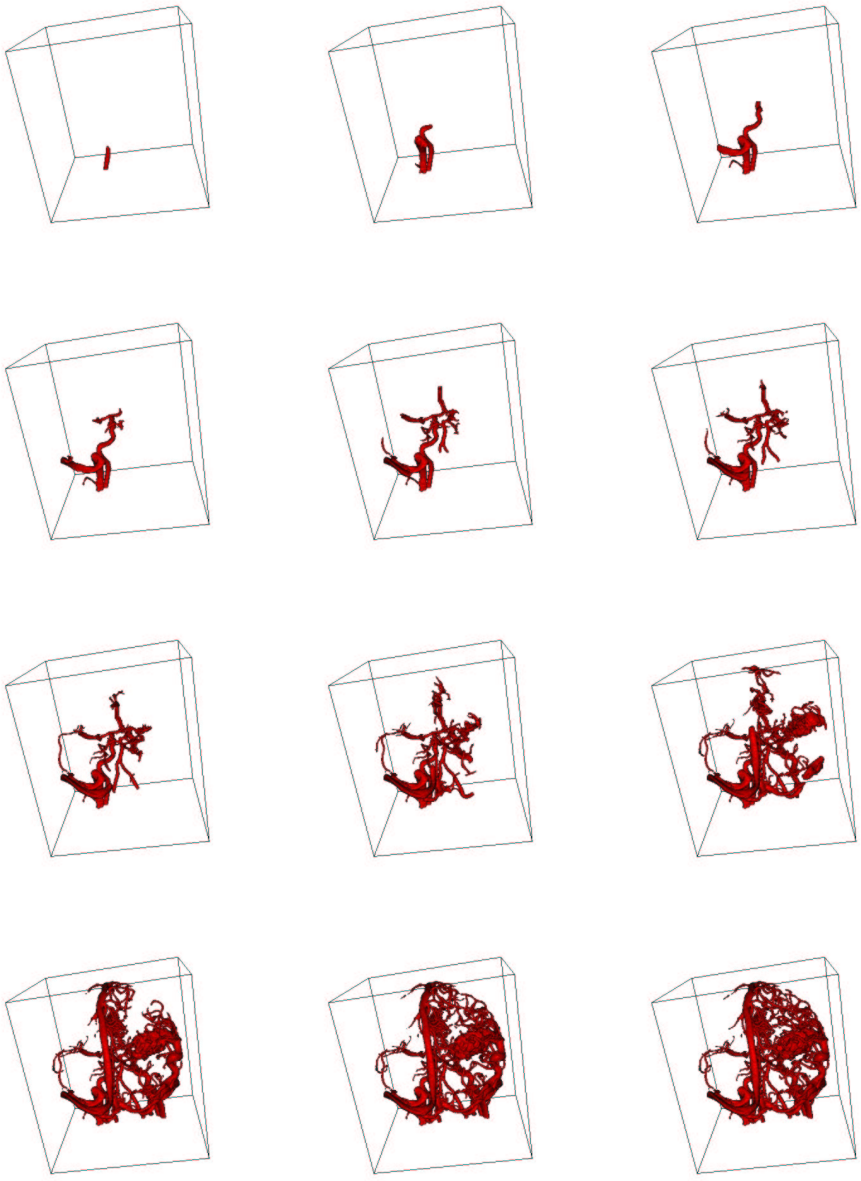


Figure 4.12. Extracting the brain vessels with region-based forces

Chapter 5

Différentes Stratégies de Segmentation Basées sur les *Level-Sets* et le *Fast-Marching*

Résumé — En dépit de beaucoup d'avantages, les *Level-Sets* souffrent de sérieux inconvénients quand on les comparent à des représentations explicites ou paramétrées des surfaces actives. Dans ce chapitre, nous allons essayer de fournir une réponse à plusieurs de ces problèmes.
En ce qui concerne le manque d'interactivité, nous montrons comment implémenter quelques techniques d'interaction avec la surface implicite en section 5.1.
Par la suite nous présentons en section 5.2 une modification des forces de "région" sur lesquelles nous nous appuierons dans les applications.
Le coût de la segmentation d'une surface peut parfois être exhorbitant, lorsqu'on rajoute une dimension au problème comme c'est le cas avec les *Level-Sets* . Nous allons utiliser le *Fast-Marching* pour fournir une initialisation rapide et précise pour que le modèle plus complexe des *Level-Sets* n'ait besoin que de quelques itérations pour converger vers une solution encore plus précise. Dans les sections 5.3 et 5.4, nous présentons un algorithme de segmentation basé sur le *Fast-Marching* et la mise en oeuvre d'un algorithme combinant nos deux méthodes en une seule.

Abstract — Despite its advantages, level-set modelization suffers from several drawbacks compared with explicit and parametric models.
No local deformations are implemented, and in section 5.1, we introduce several techniques to include interactivity in the traditional *Level-Sets* framework. We also derive variations of the region-based forces, which will be much useful for our segmentation applications.
A large number of computations is often needed to solve the variational equations involved in the *Level-Sets* model. Using the *Fast-Marching* as a segmentation tool, we are going to initialize *Level-Sets* with a pre-segmentation near the solution, where only a few iterations are needed to converge to a sub-pixel accurate solution. In section 5.3 and 5.4, we present the collaboration of *Fast-Marching* and *Level-Sets* in a single segmentation framework.

5.1 Interactive Segmentation with the Level-Sets Framework

In medical imaging no automatic method is known to be generally applicable, completely reliable and robust. As a consequence, interaction is a part of many segmentation procedure to be combined to the automated segmentation process. No interactions were introduced in level-sets models, and the effect of the parameters on the model is often non-intuitive, as shown in figure 5.1.

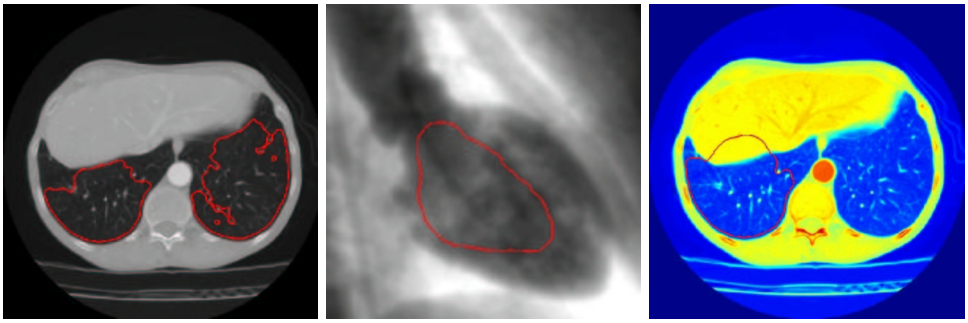


Figure 5.1. Failed segmentation with Level-Sets paradigm - Left image: The gradient influence is too important, and the level-sets is attracted by any spurious edges (vessels and arteries in the lungs); middle image: the curvature forces being too important, the inflation force is unable to reach the edges of the left ventricle; on the lung image the edges are now not sufficiently valued in order to stop inflation into the liver which surrounds the lung.

5.1.1 Interactivity requirements for the level-sets paradigm

As emphasized by most of the comparative study of explicit and implicit deformable models, and particularly in the PhD thesis of *J. Montagnat* [120], the formalism of the *Level-Sets* implicit representation restricts severely user interactivity.

The interactive requirements, when an automatic method fails, can be classified in different categories, like

- unseen evidence: the user perceive an edge where the automatic method does not (when the object intensity does not fit). In this case, the method should offer a regionally different edge defining operation or a regional parameter tuning (all examples in figure 5.1 are cases of unseen evidence);
- absence of evidence: neither the user nor the automatic method observe an edge. Interaction should imply a nullification of the evidence from the data in the region;
- dislocated evidence: the automatic method has found proof of an object according to the object model, but has confused the object with a neighboring object.

Interaction should confine the admissible contours to the indicated zone.

The definition of the correct interactivity framework is more than a difficult task, maybe more than the settle of an automatic method for medical imaging problems. The tremendous work of *S. Olabarriaga* [133, 134] on the subject settled the basis for a general interactive segmentation framework, but solutions surely lie in dedicated approaches to specific problems.

Simple interactions have been implemented in 2D for tests. They are based on modifications of the data or the model, which are two cases to answer the problems of unseen evidence and absence of evidence.

5.1.2 Fixing a point of the contour

This force will attract the zero level-set of ϕ to a user defined point. If we apply the classical evolution equation (4.5) of [135], with an added external force to the point $\mathbf{x_0}$:

$$\phi_t \quad + \quad \tilde{F}|\nabla\phi| = \phi_t + F|\nabla\phi| + \mathcal{F}_{\mathbf{x_0}}|\nabla\phi| = 0 \tag{5.1}$$

with the external force compute at each pixel $\mathbf{x}$ by

$$\mathcal{F}_{\mathbf{x_0}}(\mathbf{x}) = d(\mathbf{x_0}; \mathbf{x}) \tag{5.2}$$

This force is applied to the nearest voxels of $\mathbf{x_0}$ (in a user-chosen neighborhood), on the zero level-set. In order to compute $\mathcal{F}$ for an undefined number of fixed point $\{\mathbf{x_0}, \ldots \mathbf{x_N}\}$ in equation (5.2), we use the *Fast-Marching* algorithm, detailed in chapter 1, as explained in table 5.1. Figure 5.2 represents iterations of the evolution

Algorithm for Computing Interactivity Force $\mathcal{F}$

at iteration i, let $\{\mathbf{x_0}, \ldots, \mathbf{x_N}\}$ be the N user defined fixed points, and N action maps $U_0, \ldots, U_N$. For $j \in [1; N]$:

- $U_j(\mathbf{x_j}) = 0$, and $U_j(\mathbf{x}) = \infty$ elsewhere;
- propagate a front by computing action map U_j with $\|\nabla U_j\| = 1$ using *Eikonal equation* (1.6);
- stop when visiting a pixel $\mathbf{y}$ where ϕ sign changes.

The additional force $\mathcal{F}$ is given by $\mathcal{F}(\mathbf{x}) = \min_{j\in[1;N]} U_j(\mathbf{x})$ if $\mathbf{x}$ has been visited, in other case $\mathcal{F}(\mathbf{x}) = 0$.

Table 5.1. *Fast-Marching* for Computing external forces

process with *On-The-Fly* user interaction of a *Level-Sets* initialized by the distance to a circle, and using as evolution equation:

$$\phi_t + \mathcal{F}_{\mathbf{x_0}}|\nabla\phi| = 0 \tag{5.3}$$

where $F = \mathcal{F}$ in equation (5.1). Several comments can be added to this algorithm:

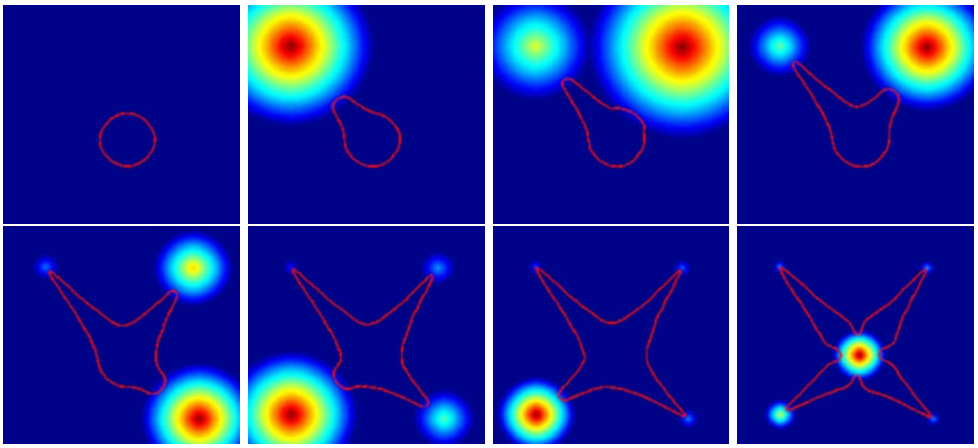


Figure 5.2. Fixing a point of the contour: these images represent the deformation of the zero level-set of ϕ, when fixing different points of the contour. The zero level-set extracted is represented in red, and is super-imposed to the external force $\mathcal{F}$ in equation 5.2

- choosing $\mathcal{F}(\mathbf{x}) = \min_{j\in[1;N]} U_j(\mathbf{x})$ instead of $\max_{j\in[1;N]}(U_j(\mathbf{x})$ in table 5.1, is just a matter of convention. The force applied is not directional, because the evolution of the curve is in its normal direction and do not integrate any tangential term. Therefore, the zero level-set is not attracted more or less by two neighboring pixels, and this is just the intensity of this force that can be tuned;

- concerning the different action maps $U_0, \ldots, U_N$, they can be integrated in only one action map U, according to the previously mentioned convention. This can greatly reduce the computing cost of N float images, for each action map. This is easily handled by taking the minimum of the different actions, because the *Fast-Marching* algorithm has been built on this minimality principle. It was the method used in figure 5.2.

5.1.3 Re-initializing the contour

The interaction we study now is a re-initialization of the contour, on the model of the *Convolution surface* developed in [14], for implicit function modeling in computer graphics (see [13]). A convolution is a modification of a signal by a filter; here the skeleton is the signal and the filter is a Gaussian kernel. More than a deformation process, this is a re-initialization of the model.

Considering our function ϕ defined on a small narrow-band, as done by *Whitaker* [184], with its *Sparse-Fields* method the level-set function is close to a binary image, as shown in figure 5.3. Defining a point $\mathbf{x_0}$ for the interaction, we compute its distance to the zero level-set, and its corresponding nearest point $\mathbf{x_1}$ on this zero level-set. In our 2D example, the segment $[\mathbf{x_0}, \mathbf{x_1}]$ is the skeleton that we convolved with a

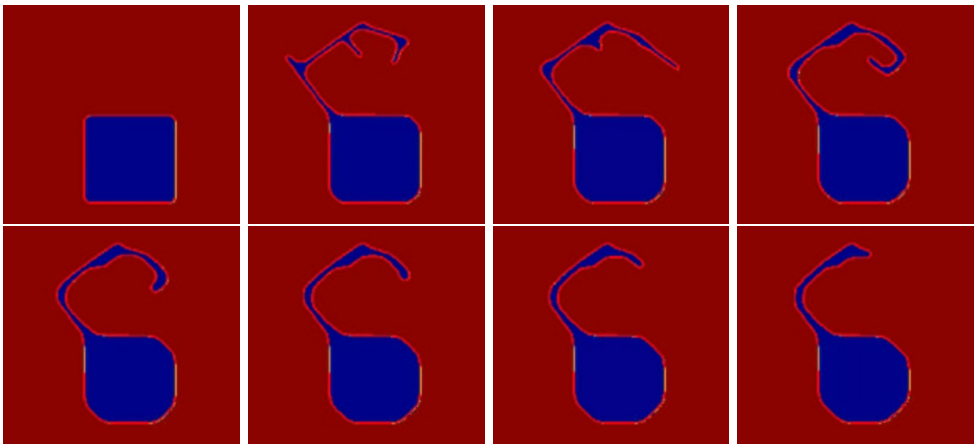


Figure 5.3. Re-initializing the contour: The initial contour which evolves according to an equation of the form $\phi_t + \kappa|\nabla\phi| = 0$ is deformed by convolution. Different steps show the curvature motion that shrinks the obtained closed curve, as explained in [70].

Gaussian kernel. We built a function $\mathcal{F}$ that for any point $\mathbf{p}$ on the image domain gives

$$\mathcal{F}(\mathbf{p}) = a\left(\int_{x_0}^{x_1} e^{-\|\mathbf{p}-\mathbf{u}\|^2/2b} du - 1/2\right) \tag{5.4}$$

that can be approximated by

$$\mathcal{F}(\mathbf{p}) = a\left(\sum_i e^{-\|\mathbf{p}-\mathbf{x_i}\|^2/2b} - 1/2\right) \tag{5.5}$$

where $\mathbf{x_i}$ are the points on the skeleton, and a, b are positive constants to control the amplitude and sharpness of the convolved object. Then we apply $\mathcal{F}$ by reconstructing the new function φ defined by the two regions:

- $\varphi^- = \phi^{-1}(\mathbb{R}^-) \cup \mathcal{F}^{-1}(\mathbb{R}^-)$
- $\varphi^+ = \phi^{-1}(\mathbb{R}^+) \cap \mathcal{F}^{-1}(\mathbb{R}^+)$

where

- if $\mathbf{x} \in \varphi^-$, $\varphi(\mathbf{x}) = \max(\phi(\mathbf{x}), \mathcal{F}(\mathbf{x}))$;
- if $\mathbf{x} \in \varphi^+$, $\varphi(\mathbf{x}) = \min(\phi(\mathbf{x}), \mathcal{F}(\mathbf{x}))$.

Applying this method to ϕ, we modify the zero level-set as done in figure 5.3. This implementation of a re-initialization is rather similar to the method proposed in [141].

5.1.4 Conclusion on the interactivity

We have shown two possible interactivity techniques, based on a modification of the data, and on a modification of the model, both based on computer graphics ideas. Basically, a lot of techniques first studied for computer graphics can be applied to the *Level-Sets* which basis is an implicit representation. The implicit representation in computer graphics (see [13]) has much in common with the formalism developed in [135]. It was even studied by *Cani and Desbrun* in [19] and detailed in [41], with a *Level-Sets* implementation for a "skin" that contains spherical particles for the simulation of highly deformable models. But in Computer Graphics, first matter is the rendering of animations, and not their computing costs. Unfortunately, interactivity requires a direct and fast output of any user interaction, which is something that is still difficult to manage with the *Level-Sets* paradigm.

But the solution probably lies in the combination of two different techniques: one for segmentation, and one for interaction. A very good example of this philosophy can be found in [188]: the authors use circular oriented particles to sample and control implicit surfaces. This technique is also used by *Szeliski et al.* [171]. The model is a force-feedback system where particles try to match the zero level of the implicit function, and where the surface follows particles. A simple constraint locks the set of particles onto a surface while the particles and the surface move. The particles use mutual repulsive forces and fissioning to sample the whole surface, and the user interactivity can be applied locally on one particle at a time, using them as control points for direct manipulation, as shown in figure 5.4[1]. Finally, very interesting work

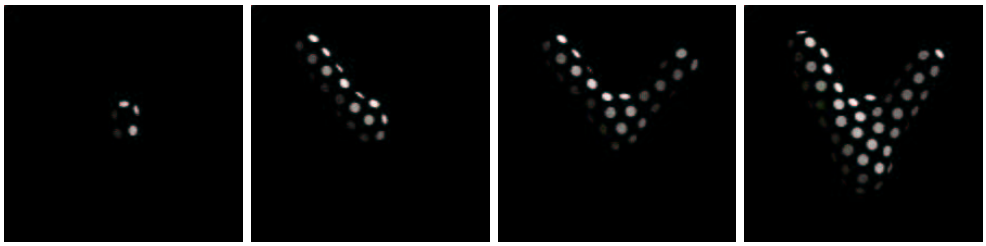


Figure 5.4. Interactivity on an implicit surface : the particles track the implicit surface, leading to an easy visualization of its zero level-set, and at the same time input a control on this surface, in order to modify its shape.

has been done on the construction of subjective contours in [154], where the *Level-Sets* extracts contours in famous test images of *Kanizsa* [81] that strongly requires image completion.

[1]We would like to acknowledge Andrew Witkin and Paul Heckbert, for providing this sequence that is an illustration of their article [188].

5.2 Region Based Forces

5.2.1 Gaussian descriptors

We recall the descriptors first introduced by *Zhu and Yuille* [196]. They are "fully automatic" in the sense that no user-defined parameter is necessary to their definition. They are based on region probabilities defined by Gaussian distributions:

$$P_{in}(x,t) = \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)}\ \exp(-\frac{(I(x)-\mu_{in}(t))^2}{2\,\sigma_{in}^2(t)})$$

A similar expression is used for the definition of $P_{out}(x,t)$. Here $\mu_{in}(t)$ and $\sigma_{in}^2(t)$ are respectively the mean and the variance of the image intensity over $\Omega_{in}(t)$:

$$\mu_{in}(t) = \frac{\int_{\Omega_{in}(t)} I(x)\,dx}{\int_{\Omega_{in}(t)}\,dx}\ ,\ \sigma_{in}^2(t) = \frac{\int_{\Omega_{in}(t)}(I(x)-\mu_{in}(t))^2\,dx}{\int_{\Omega_{in}(t)}\,dx}$$

This means that the histograms of the intensity in $\Omega_{in}(t)$ and $\Omega_{out}(t)$ are *modeled* by Gaussian distributions.

5.2.2 Modified Gaussian descriptors

The preceding model has one major drawback: if one of the variances is much smaller than the other one, then undesired results can be observed. For example, if the image is composed of a bright region with a small variance (a contrast-enhanced organ for instance) in the middle of a dark background with a large variance, then very bright pixels may have a greater probability to be in the background than in the bright region. This is perfectly normal from the point of view of the Gaussian model, but it is in contradiction with our *a priori* knowledge about the image informational content. In other words, the Gaussian model may be unadapted to some images.

In such cases, we can introduce an priori knowledge by modifying the two Gaussian distributions. If we know that the image region that we want $\Omega_{in}(t)$ to cover is supposed to be brighter than the one covered by $\Omega_{out}(t)$, then we take:

$$P_{in}(x,t) = \begin{cases} \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)}\ \exp(-\frac{(I(x)-\mu_{in}(t))^2}{2\,\sigma_{in}^2(t)}) & \text{if}\quad I(x) < \mu_{in}(t) \\ \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)} & \text{if}\quad I(x) \geqslant \mu_{in}(t) \end{cases}$$

and:

$$P_{out}(x,t) = \begin{cases} \frac{1}{\sqrt{2\,\pi}\,\sigma_{out}(t)} & \text{if}\quad I(x) < \mu_{out}(t) \\ \frac{1}{\sqrt{2\,\pi}\,\sigma_{out}(t)}\ \exp(-\frac{(I(x)-\mu_{out}(t))^2}{2\,\sigma_{out}^2(t)}) & \text{if}\quad \mathcal{I}(x) \geqslant \mu_{out}(t) \end{cases}$$

It is also possible to manually choose the means and variances instead of computing them from the region histograms, which can be very useful in some applications (they can be obtained through an initialization process). In this case, the region descriptors become time-independent.

5.2.3 Sigmoid descriptors

For the images where the modified Gaussian model failed, we used user-defined time-independent sigmoid probabilities. We took:

$$P_{in}(x,t) = \frac{1}{1+e^{a_{in}\,(b_{in}-I(x))}} \tag{5.6}$$

with a similar expression for $P_{out}(x,t)$. While Gaussian have quadratic logarithms, sigmoid functions have asymptotic linear logarithms. For example, if a_{in} is positive, then:

$$\lim_{I(x)\to+\infty} \log P_{in}(x,t) = 0$$

and:

$$\log P_{in}(x,t) \sim_{I(x)\to-\infty} a_{in}\, I(x).$$

Those descriptors for the aorta of figure 3.12 are shown in figure 5.5.

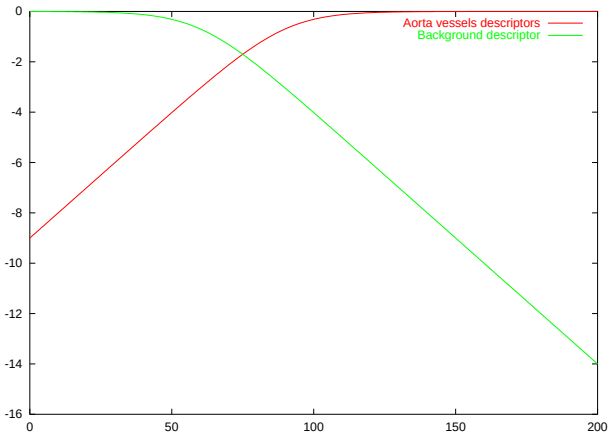


Figure 5.5. Sigmoidal region-based forces: they are used for discriminating the aorta towards the background in figure 3.12.

Figure 5.6 shows iterations of the segmentation of the aorta of figure 3.12 with sigmoidal region descriptors of figure 5.5.

5.2.4 Numerical implementation of the level-sets paradigm

Adaptive time step

We used an Euler first-order explicit time scheme for solving (4.6). We recall that explicit time schemes often require a limitation of the time step to be stable. Here it is possible to control the Courant-Friedrichs-Lewy (or CFL) number[2] of the discretized flow equation, by adapting the time step between each time iteration.

[2]Number of cells crossed by a characteristic curve during a time step.

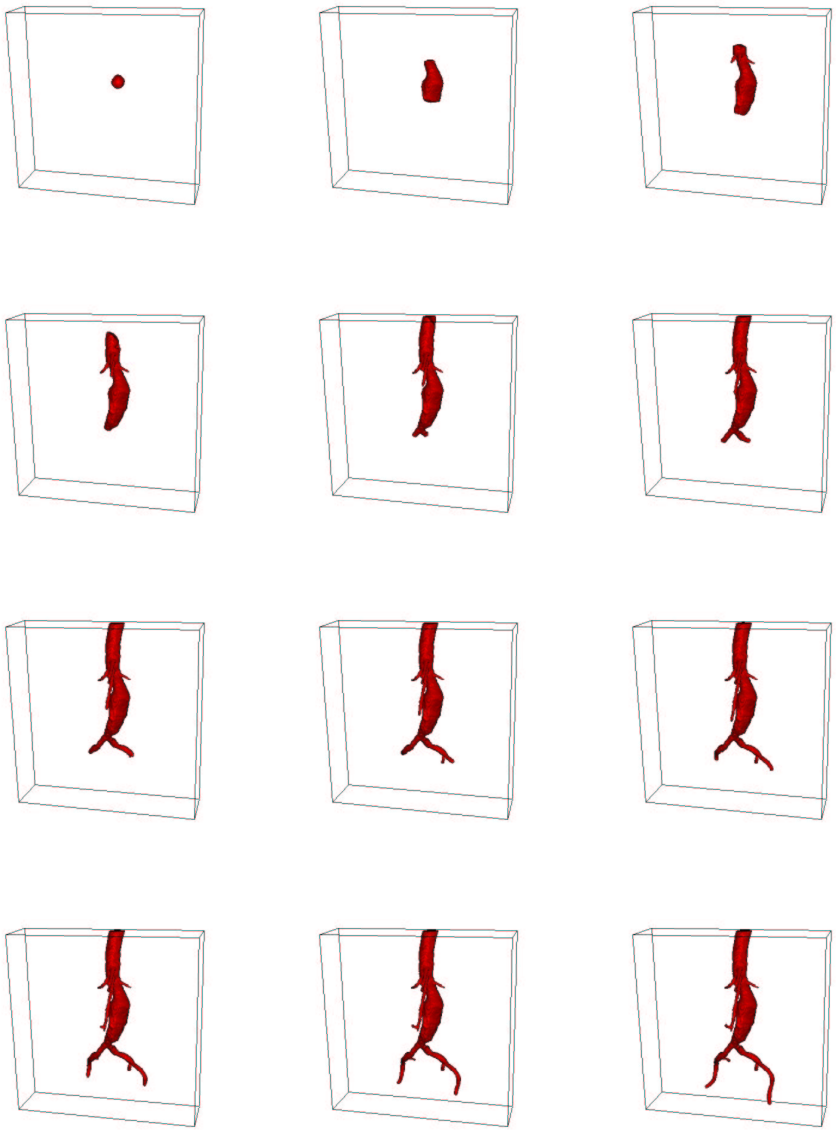


Figure 5.6. Extracting the aorta with region-based forces: the initialization is a sphere located inside the aorta object. Using the *a priori* distribution of the grey levels inside the aorta and boundary based forces, we segment the aorta, as shown on the several iterations.

The only difficulty is related to the scheme associated with the curvature flow, which has no obvious stability conditions. We have empirically chosen to give a stronger penalty to its contribution to the global CFL number. Explicitly in 3D, for a composite flow of the following form:

$$\mathbf{V} = (\beta(\mathbf{x},t) - \varepsilon(\mathbf{x},t)\,\kappa_M(\mathbf{x},t))\mathbf{n} + \mathbf{U}(\mathbf{x},t)$$

we ensure that:

$$\max_{(i,j,k)} (\nu^{\beta}_{ijk} + 3\,\nu^{\varepsilon}_{ijk} + \nu^{U}_{ijk}) \leqslant 1$$

where:

$$\begin{aligned}
\nu^{\beta}_{ijk} &= \Delta t\,|\beta_{ijk}|\,\sqrt{\Delta x^{-2} + \Delta y^{-2} + \Delta z^{-2}} \\
\nu^{\varepsilon}_{ijk} &= \Delta t\,|\varepsilon_{ijk}|\,\sqrt{\Delta x^{-2} + \Delta y^{-2} + \Delta z^{-2}} \\
\nu^{U}_{ijk} &= \Delta t\,\left(\frac{|U_{x\,ijk}|}{\Delta x} + \frac{|U_{y\,ijk}|}{\Delta y} + \frac{|U_{z\,ijk}|}{\Delta z}\right).
\end{aligned}$$

The results are very satisfying, and no numerical instability has been noticed during the segmentation tests.

Narrow Banding

For reasons of causality, it is possible to restrain the computation domain to a band of cells around the zero-level set of $\phi(\cdot,t)$. The result is a decrease of the computational cost. Several approaches have already been proposed, but we have build a new one which is adapted to our segmentation problems.

Classical approaches are referred to as narrow-band methods (see figure 5.7). Some are based on combinatory studies in order to add or remove cells around $\phi(0,t)$ as it moves (see *Adalsteinsson and Sethian* [2]). Another approach can be found in the thesis of *Keriven* [84], and consists in computing the signed distance function to $\phi(0,t)$ when it gets too close to the boundaries of the band, and re-initializing $\phi(\cdot,t)$ with the computed distance function. Other authors chose to initialize ϕ with a distance function and maintain $\|\nabla\phi\| = 1$ at each iteration (see *Adalsteinsson and Sethian* [3], and *Gomes and Faugeras* [68]).

These methods cited above are efficient, especially when the initial condition $\phi(\cdot,0)$ is far from the solution. Here we propose a new specific method that is very efficient for the segmentation of fine- to medium-size tubular structures (like arterial, venous, or bronchial trees), or when using a rough pre-segmentation for initializing ϕ.

The idea is simple: we start from a distance function, either associated to a pre-defined geometric primitive, or computed from a binary pre-segmentation by a fast-marching algorithm (see next chapter). We define a narrow-band area and a wide-band area around $\phi(0,0) = \phi_0$ (the wide-band is larger than narrow-band and comprises it). The calculations are then performed in every cell of the wide-band area. If $\phi(0,t)$ gets out of the narrow-band area (i.e. the sign of ϕ changes somewhere

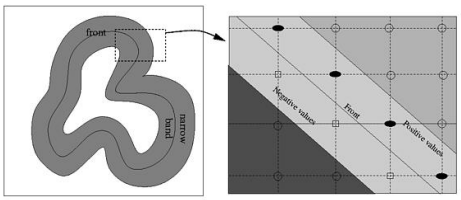


Figure 5.7. Narrow Band Diagram: Left image shows the overview of the narrow-band structure, and right image shows the detail.

Figure 5.8. Narrow Band reconstruction: images are samples of the evolution and reconstruction of the narrow-band, starting from two circles that merge.

in the outside of the narrow-band area), then both bands are locally enlarged in a spherical manner.

As a consequence, we can add cells to the computational domain, but we never remove cells from it. If a pre-segmentation is used, the enlargements of the bands are few and the resulting computational cost is very small. This method has also proved to be efficient in the case of propagation in tubular structures with diameters comparable to the wide-band width (for example 10 times a cell diagonal).

Boundary conditions

We used so-called *free* boundary conditions, that is to say that we extrapolate the values of $\phi(\cdot, t)$ in the outside vicinity of the computation domain boundaries by taking the closest value inside the domain, and that the corresponding finite differences are equal to zero. Some authors use other entropic boundary conditions. The main idea is that the information that goes out of the computation domain is lost, and that no information should be introduced into the domain through its boundaries.

5.3 Using the Fast-Marching for segmentation

Originally introduced by *Malladi and Sethian* [111], the *Fast-Marching* can be used as a fast initialization algorithm for image segmentation. Despite the fact that it requires low computational cost, it has several drawbacks, in particular that the speed F in equation (4.20) is always positive or always negative. Therefore, the front evolves and propagates in the whole image domain, without being stop by any edge, and it cannot be used for cases with curvature-dependent speed functions. The stopping criterion still relies on a user perception of the segmentation, like visual inspection of the domain visited by the algorithm during propagation, or the choice for an appropriate stopping time, as done in [111]. But the nice segmentation properties of the *Fast-Marching* algorithm shall be taken into account, as shown in figure 4.10, where we segment a brain on the basis of the method described in [111].

Still, the stopping criterion in this case was to visually control the segmentation process, because the automatic detection of the crossing of the interesting edges is difficult to implement.

Noticing that *Eikonal equation* is used to model the continuous formulation of the watershed transform, we have developed an interactive segmentation tool based on the flooding process of this morphological algorithm introduced in [180]. It is based on the same principle that segmentation boundaries are defined by pixels where several evolving fronts collide.

5.3.1 Comparison with the watershed algorithm

The classical morphological segmentation paradigm [156] is based on the watershed transform [181]. The method is very simple:

- the gradient image $\|\nabla I\|$ is computed (and enhanced);
- for each object of interest, an inside particle is detected (either interactively or automatically);
- flooding waves are propagated from the set of markers and flood the topographic surface $\|\nabla I\|$.

The points where the flooding waves meet each other form the segmentation boundaries. Different regions between boundaries are called catchment basins. Usually markers are the regional minima of the gradient image. An example of the watershed transform is shown in figure 5.9. Very often, the minima are extremely numerous, leading to an over-segmentation, therefore for practical cases, the watershed transform will take as seed points a smaller set of markers, identified through pre-processing techniques.

The flooding waves are propagated according to the topographic map $\|\nabla I\|$, where the pixels belonging to a catchment basin are nearer to its corresponding regional minimum than to any other minima. In this framework, the construction of catchment basins can be seen as a shortest path problem between a marker and the image points. It corresponds to compute the gray-weighted distance transform (GWDT) of the image to the set of markers. There are several ways to compute this $GWDT$:

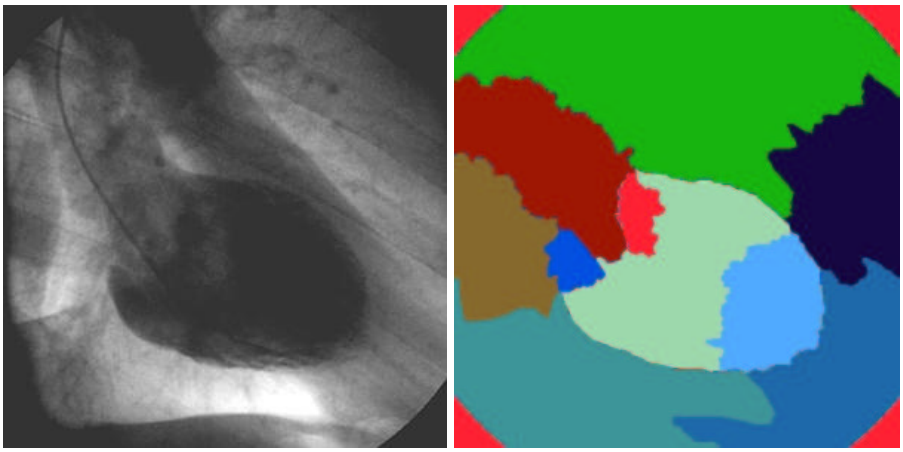


Figure 5.9. Watershed transform of a 2D X Ray image of the heart (LV): The seed points are regional minima of the gradient of image 5.9-left; the flooding was implemented in the continuous framework of *Eikonal equation* , and the regional minima were filtered in order to have a smaller set of markers.

as shown in [118], viewing the topographic image domain as a refractive index, as explained in chapter 1, the distance between a marker and any point in the image is the line integral of the penalty $\|\nabla I\|$ along the optical path length. We reformulate *Eikonal equation* 4.22 with

$$\|\nabla T(\mathbf{x})\| = \|\nabla I(\mathbf{x})\| \tag{5.7}$$

and the *GWDT* can be computed on the whole image domain, using the markers as sources pixels p where $T(\mathbf{p}) = 0$.

Maragos has shown in [118] that the continuous segmentation approach based on the *Eikonal equation* outperforms the discrete segmentation results. However several problems remain important:

- the final segmentation relies on the number of the markers: too many markers lead to an over segmentation of the image, and merging algorithms are needed as post-processing;
- it relies also on the quality of the edge detector applied to the real image: if there is a gap in the edge information, the nearest marker will flood the adjacent region.

We have devised a very simple method based on user interactivity, and different front speeds, based on the formulation of the flooding of the whole image. Each front, initialized by a user-defined seed point, will propagate according to its own propagation speed, derived from local image information (and not just gradients), and will stop when it meet other propagating fronts. This method has been implemented to provide a *quick and dirty* initialization for the *Level-Sets* method.

5.3.2 Interactive segmentation with the Fast-Marching

Flooding algorithm

In the following, we assume that we have an undefined number of seed points for front propagation, that can be labeled with a known number of labels. We detail below the flooding algorithm for multi-label front propagation.

Definition

- a set of starting point $\mathbf{p}_1, \ldots, \mathbf{p}_n$, located inside the image domain;
- each starting point $\mathbf{p}_i$ is assigned a label l_i $i \in [1; n]$ (not necessarily different);
- a label map $\mathcal{L}$, for controlling collision;
- a penalty image $\mathcal{P}$ which will drive the front propagation, and which is a function of the position and the label of the front;
- We initialize the usual set of data-structures for front propagation, including an action map $\mathcal{A}$ and (only) one min-heap structure $\mathcal{H}$;

Initialization

- we initialize a classical front propagation method, setting $\mathcal{A}(p_i) = 0$ $\forall i \in [1; n]$ and storing all seed points in the min-heap structure;
- $\mathcal{L}(\mathbf{p}_i) = l_i$, $\mathcal{L} = -1$ elsewhere;

Loop: at any iteration

- Let $\mathbf{x}_{min}$ be the *Trial* point with the smallest action $\mathcal{A}$;
- Move it from the *Trial* to the *Alive* set (i.e. $\mathcal{A}_{\mathbf{x}_{min}}$ is frozen);
- Update $\mathcal{P}(\mathcal{L}(\mathbf{x}_{min}))$
- For each neighbor $\mathbf{x}$ (6-connexity in 3D) of $\mathbf{x}_{min}$:
  - If $\mathbf{x}$ is *Far*
    1. add it to the *Trial* set;
    2. $\mathcal{L}(\mathbf{x}) = \mathcal{L}(\mathbf{x}_{min})$
    3. compute $\mathcal{A}$ according to equation

$$\|\nabla \mathcal{A}\| = \mathcal{P}\left(\mathcal{L}(\mathbf{x}_{min})\right) \tag{5.8}$$

       using the up-wind discretization scheme, involving only the neighbors of $\mathbf{x}$ with label $\mathcal{L}(\mathbf{x}_{min})$;
  - If ($\mathbf{x}$) is *Trial*
    1. recompute the action $\mathcal{A}(\mathbf{x})$ with equation(5.8), involving only the neighbors of $\mathbf{x}$ with label $\mathcal{L}(\mathbf{x}_{min})$;
    2. if the new value is smaller, then $\mathcal{L}(\mathbf{x}) = \mathcal{L}(\mathbf{x}_{min})$

Termination

- stop when all pixels on the image domain are visited;
- the label map $\mathcal{L}$ represents the final segmentation into k regions R_k.

In this algorithm, we have deliberately left blank the update procedure of the potential. Using a very simple potential based on the grey level information. Having the n seed points $\mathbf{p}_1, \ldots, \mathbf{p}_n$, we can set at initialization

$$\text{if } \mathcal{L}(\mathbf{x}) = \mathcal{L}(\mathbf{p}_i) \ , \ \mathcal{P}(\mathbf{x}) = \max(I(\mathbf{x}) - I(\mathbf{p}_i), 0) + w \tag{5.9}$$

Then the speed is inversely proportional to the difference between the grey-level of the starting seed point and other points in the image. Result of this strategy can be observed in figure 5.10 where different seed points were manually located on the image 5.10-left, leading to the segmentation of figure 5.10-right. The resulting is not

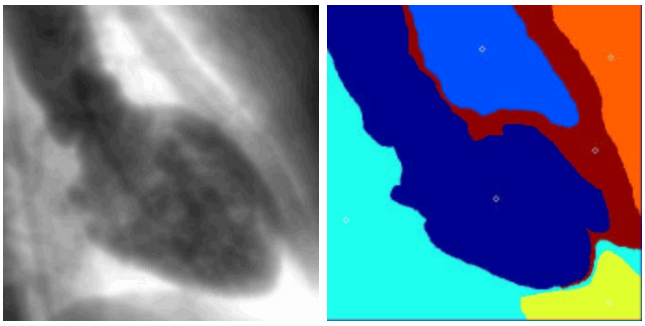


Figure 5.10. Front competition on a 2D X Ray image of the heart (LV): The seed points were manually located on the dataset on the left image; they are visible as white dots on the right image.

convincing, but the use of this kind of information to differentiate the different front speeds has a bigger potential than the classical watersheds formulation. If there is a gap in the gradients between two basins, one of the two fronts could flood completely into the adjacent basin with the watersheds, whereas the speed in the adjacent region could still be discriminating the unwanted flooding, in the case of our competitive fronts algorithm. This is mainly the difference between a difference of a *plateau* and a barrier in the penalty (where *plateau* is a region with small variations in the intensity).

Thus, different strategies can be implemented, as many as there are different possible potentials.

Setting the penalty information

We can devise a very simple algorithm that do not use the information coming from the seed points, but for the m first visited voxels for each label. This reduces greatly the influence of the user interaction in setting the markers position. Therefore for each label,

- At iteration i in the *Fast-Marching* we examine the point $\mathbf{x}$ in the min-heap whose action is minimal;
- We remove it from the heap and we examine its label $l = \mathcal{L}(\mathbf{x})$ and its grey level $I(\mathbf{x})$;

- For the front points with label l, we know N_l the number of points considered and μ_l the mean grey level value of those points;
- if $N_l < m$, we update this mean grey level value for the label l

$$\mu_l = (N_l * \mu_l + I(\mathbf{x}))/(N_l + 1) \ , \ N_l + + \tag{5.10}$$

- considering that the image grey level of the pixel in the desired region are a random distribution of mean μ_l we can also update the variance with *Koenig-Huyghens* formula $\sigma_l^2[X] = \mu_l[X^2] - \mu_l[X]^2$ for the random variable X;
- we use this new mean values μ_l and σ_l in the computations of the action at each neighbors of $\mathbf{x}$.

In figure 5.11 is shown an example of using this strategy for setting the penalty term. The speed function in the image shown is computed with $m = 10$ for all

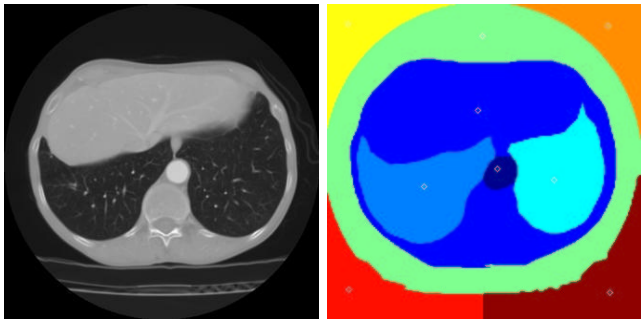


Figure 5.11. Front competition on a 2D scanner slice of the lungs: The seed points were manually located on the dataset on the left image; they are visible as white dots on the right image.

labels. The speed function for each label l is computed with the penalty $\mathcal{P}(\mathbf{x}) = e^{-|I(\mathbf{x})-\mu_l|/2\sigma_l^2}$.

Unfortunately, this algorithmic trick makes the penalty a time-dependent function, and thus the minimality principle is violated, and *Eikonal equation* discretization cannot be applied in theory during the iterations where $\mathcal{P} = \mathcal{P}(\mathbf{x}, t)$. Therefore, this heuristic must be seen as a region growing algorithm to operate a statistical study at the beginning of the segmentation process before visiting the whole image domain. It can also be replaced by a similar statistical study in a sphere around the seed points.

Figure 5.12 displays the result of the same strategy on the brain dataset used for test in figure 4.10. This result was obtained by setting different seed points in the different objects in the brain MR image which is represented on the left column of figure 5.12. Each one of the regions was initialized with one marker, and they are represented when all fronts collide super-imposed on three orthogonal views of the dataset in the left column of figure 5.12. Inconvenient of this kind of segmentation is that it assumed that the correct number of classes is known *a priori* and that the user can clearly indicate to the algorithm the location of each connected components

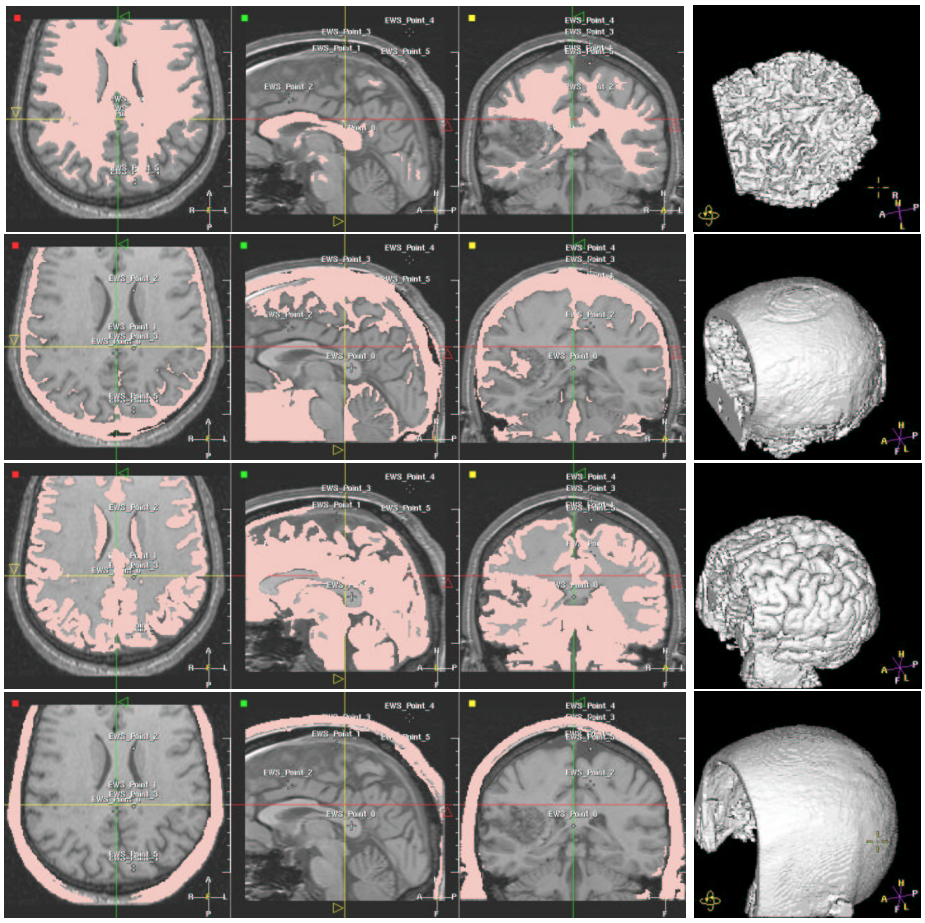


Figure 5.12. Front competition result on the brain: Simultaneous segmentation of the white and grey matter of the brain, plus the skull and the skin; each row corresponds to a region, and the intersection between the regions and the datasets is represented in pink, in the left column.

of this class. But target here is to provide a quick and dirty initialization to a more complex and time consuming method (like *Level-Sets*).

The same test was done on a lung image, where we try to separate the airways from the soft tissue and vessels, in figure 5.13. Results The underlying dataset is a volume-of-interest extracted from a multi-slice CT scanner of the lungs. The dataset, almost isotropic, contains a huge number of pixels, and even a fast algorithm like ours is not able to achieve this result in "interactive time". But we have shown the ability of this simple algorithm to achieve difficult segmentation tasks.

Further, in this algorithm, there is no need to use several data-structures (like min-heap) to store each different front. Due to the minimality principle of the *Fast-*

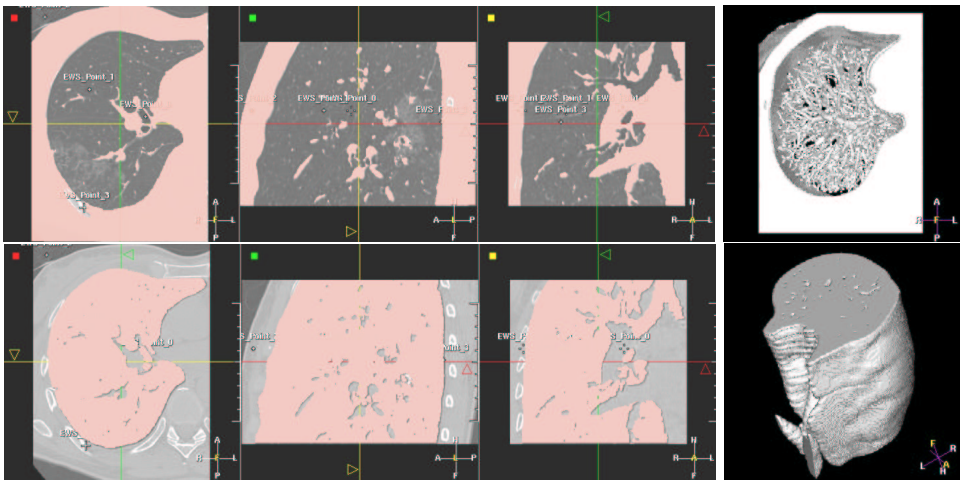


Figure 5.13. Front competition result on the lungs: separating air and soft tissues in a multi-slice CT dataset of the Lungs; seed points are manually located in the different parts of the image (on in the airways, on in the tissues, and on outside the object).

Marching construction, the point with minimum energy, independently from its label, will still be at the root of the min-heap data-structure. And each point of the image will be visited only one time, leading to approximately the same computing cost than the propagation of a single front in the image domain. It seems that a similar version of the multiple front propagation algorithm has been developed in [103, 167], for initializing color and texture segmentation with the *Level-Sets* methods.

5.4 Combining Fast-Marching and Level Sets

5.4.1 Advantages and Drawbacks of *Fast-Marching*

The *Fast-Marching* algorithm is used for the computation of a minimal action map, or weighted distance transform (from the morphological point of view).

Drawbacks : monotonicity of the speed (can only propagate in one direction), no control of the curvature of the contour to be extracted, no stopping criterion (speed always strictly positive/negative).

Advantages

- It is a very fast algorithm (as shown in the case of path extraction for virtual endoscopy in chapter 3);
- it can segment objects with complex topologies (see the brain example in figure 4.10);

- it can be used with simple initialization of seed points (see the competitive front segmentation of figure 5.12);
- There are lots of Penalty definition (depending on user's imagination, ranging from grey-levels to Hessian measures [60], and other complicated measured obtained through filtering).
- it can be used for initialization: Figure 5.14 shows images of the conversion from an implicit representation of the object, to an explicit model.

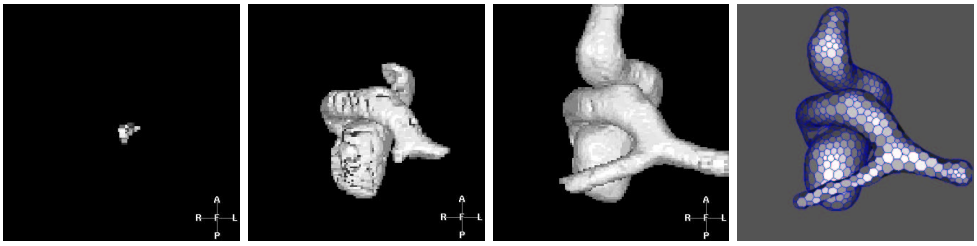


Figure 5.14. Combining Fast-Marching with Explicit Methods : the first three images are samples of the segmentation process described in section 5.3; the right image is the conversion of the implicitly defined model into a simplex mesh.

Drawbacks

- Monotonicity of the speed (can only propagate in one direction);
- No Curvature term in the Energy of the contour to be extracted;
- No stopping criterion (Speed always strictly positive/negative).

5.4.2 Advantages and Drawbacks of *Level-Sets*

Level-Sets implementation allows to evolve embedded curves with wide variety of force models, and lead to high-accuracy segmentations with sub-pixel precision. But they often use time-consuming iterative approaches. They have always been criticized because of their computational cost. For instance being used for the segmentation of the cortex (see for example *Zeng et al.* [195]), they provide very satisfying results, but several hours of computation are needed. Several implementations try to override this problem.

The speed of the algorithm has been increased with the use of semi-implicit discretization schemes by *Goldenberg et al.* [65, 67]. This method is based on the implementation of a semi-implicit discretization scheme originally applied to non-linear diffusion filtering [182]. Only drawback is the assumption that the slope of the *Level-Sets* guarantees $\|\nabla\phi\| = 1$ across iterations, with an evolving equation based on the

formulation of [23]. This condition is not met with the *Level-Sets* formulation, but another model developed in [69] overcomes this drawback.

An interesting algorithm was derived in [137]: the *Hermes* algorithm is based on the idea to propagate the pixel that evolves faster at each step, thus combining the *Narrow-Band* and *Fast-Marching* approach, with propagation over a relatively small window. However, the proposed algorithm does not solve exactly the given partial differential equation since the evolution is computed locally, and this acceleration step is not valid in an homogeneous media, where the whole curve is propagating with the same speed (as shown for a pure advection flow in figure 4.6).

5.4.3 Combining two complementary methods

In the domain of medical image analysis, time-consuming algorithms are synonymous with non-interactive methods, and are therefore limited to a very small number of specific applications. In others words, the computation times of level sets are an obstacle to a wider use of these methods. The strategy we used is an original approach which tries to go beyond this classical barrier.

Both methods have a common advantage : they have no constraints or hypothesis on topology, which may change during convergence.

Our segmentation strategy will consist in using *Fast-Marching* method as a pre-segmentation tool, and then refine the segmentation with a level set method. This approach combines the advantages of both methods: the fast-marching pre-segmentation is rough but quick, and the level set needs only a few iterations to produce the final, highly accurate segmentation.

By combining fast-marching and level sets, we build a tool which is able to produce highly accurate segmentations of topologically and geometrically complex structures in minutes where level sets alone took hours. The framework is the following

Fast-Marching will provide a fast and rough initialization, near the solution. As a second process it will give a statistical study of the pre-segmentation (for the level-sets forces): looking at the local distribution of the different regions extracted with the competitive front, we are able to give the initial values of the region descriptors. Computing the mean and variance of different regions, we can skip the **MDL** statistical study and assume that those values are the original first and second moments of the Gaussian probabilities in section 5.2. And *Level-Sets* will start from the segmentation step achieved by *Fast-Marching* . In a few iterations, they will converge with more complex external forces, including region-based terms as in sections 4.4 and 5.2. Their particular formulation will lead to an accurate position of the sub-pixel iso-surface, that enhances visualization and measurements of pathologies.

Chapter 6

Applications de la Combinaison des Techniques de *Fast-Marching* et de *Level-Sets* à l'Extraction de Surface

Résumé — Dans ce chapitre, nous avons étudié deux problèmes où la mesure et la visualisation du résultat de la segmentation sont primordiales.
Tout d'abord, on a étudié les anévrismes du cerveau, qui sont des gonflements des artères, qui peuvent mener à une hemorragie cérébrale et plonger le patient dans le coma. Dans ce cas, c'est la précision et la rapidité de la segmentation qui sont des éléments fondamentaux pour préparer une intervention chirurgicale.
Le second problème est celui des polypes du colon. Dans le cas de la colonoscopie virtuelle de la section 3.1, l'utilisateur regarde l'intérieur du colon, et la détection des polypes repose entièrement sur ses indications. Nous avons utilisé notre méthode de segmentation pour automatiser et rendre robuste cette tâche de détection en améliorant la visualisation.

Abstract — In this chapter, we have studied two different problems related to typical pathologies, where measurements and visualization are the main objectives. First problem is the segmentation of cerebral aneurysm which are dilation of the brain vessels that may burst and lead to coma. Accuracy and speed of the segmentation process are needed in order to prepare interventional treatment. We therefore applied successfully our methods to segment and extract a shape representation of cerebral aneurysms.
Second problem is the already studied colon polyps visualization. In virtual colonoscopy, as developed in section 3.1, the user observe the interior of the colon, and detection is based on its indications. We want to automate this task, and we give preliminary results on an interesting visualization mode that could lead to this target.

6.1 Segmentation of cerebral aneurysms

What is an aneurysm?
An aneurysm is an abnormal dilatation involving the wall of an artery, a thick-walled blood vessels that carry blood pumped from the heart, under high pressure. Aneurysms can develop on any artery within the body. Common arteries where aneurysms are found include the aorta, the popliteal artery (behind the knee) and brain arteries. We focus on the brain aneurysms (see figure 6.3-right). A brain aneurysm begins as a small thinned area at the wall of an artery at the base of the brain (see figure 6.1). Over time the blood flow pounds against the thinned portion

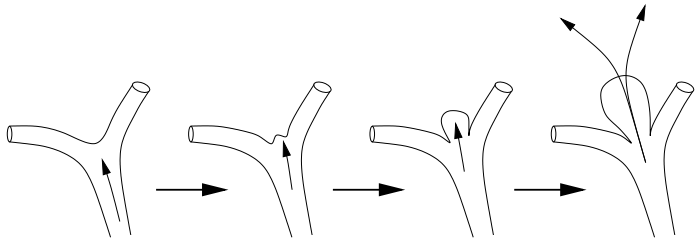


Figure 6.1. Development and rupture of a brain aneurysm: The blood flow, represented by an arrow, stresses an area of "potential" weakness at the branching point of an artery. Over time, this point of weakness dilates into an aneurysm. The blood enters the aneurysm itself and finally escapes from a rupture point at the top of the aneurysm.

of the wall, and eventually it starts to dilate, inflating like a balloon, creating the aneurysm, and as it grows, the wall of the artery gets thinner and thinner, until it ruptures.
What causes an aneurysm to form?
This is still an open question. Aneurysms arise at an area where the wall of an artery is thin. Most arteries in the body have walls with three layers, and brain arteries have segments where one of the layer is absent, which can contribute to the problem. An aneurysm may be caused or aggravated by disease such as hypertension, because it results in high blood pressure, or may be caused by any disease which affects the walls of the arteries (even smoking).
What dangers do aneurysms present?
The danger from an aneurysm is that it will continue to bulge and may burst. When an aneurysm in a large blood vessel or in the heart bursts, a person could bleed to death. When an aneurysm bursts in the brain, a stroke (brain attack) can result.
How are cerebral aneurysms diagnosed?
Biplane angiography has become a standard imaging procedure for the treatment of cerebral aneurysms. Recently, there has been a lot of interest in 3D visualization of intracranial vessels in interventional neuroradiology [95]. A clinical application of 3D-Rotational Angiography (**3D-RA**) has been developed by Philips, using a standard angiographic system, with a C-arm that performs a rotational angiographic

acquisition around the patient, and provide accurate 3D reconstruction [45]. With classical rendering techniques, it enables the clinician to observe for example the relationship of a parent vessel with the neck of an aneurysm, in the brain vessels. The availability of three-dimensional information during the intervention increase the possibilities towards a more accurate and time efficient endovascular treatment. With volume rendering tools, the clinician is able to see structures from any angle. But still, it relies on threshold-based visualizations techniques.

How are cerebral aneurysms treated?
There are two ways to treat an aneurysm: First one is surgical treatment, where the clinician places the patient under general anesthesia. A window is opened in the skull bone of the patient head. When the aneurysm is in view, it is separated from the surroundings structures by dissection, and a metal clip is closed across the base of the aneurysm. Therefore blood no longer flow into the aneurysm (see figure 6.2-middle). Second way is to do endovascular coiling: this technique consists of filling the

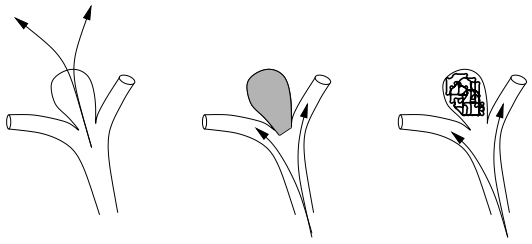


Figure 6.2. Treatment of a brain aneurysm: Left image shows an aneurysm at a branch point; middle image illustrates the placement of a clip across the base of an aneurysm correctly placed; right image illustrates the packing of the same aneurysms with coils.

aneurysm from the inside with a long length of fine platinum wire that coils around and around inside the aneurysm, until the blood flow stops entering the aneurysm (see figure 6.2-right). In order to reach the aneurysm, the clinician puts a fine catheter through the artery, and advances it into the head until it reaches the aneurysm. Then the wire is passed up into the catheter and into the aneurysm, until it is full of wire. This treatment has its own risks and complications, since it is still a very invasive technique. The aneurysm can burst during the treatment. In order to perform this examination in the best condition, the clinician needs to output from the angiography an accurate model of the aneurysm. The surface of the object segmented can be used to do measurements, but also flow simulations in order to determine critical points of possible rupture.

We have applied the segmentation models used in section 5.4, in order to increase the accuracy of the visualization of the pathologies. Increasing accuracy and speed of the computations is important for a system in an interventional environment. The 3D shape information will enhance analysis of the structures, measure of the vascular anatomy, and preparation of the treatment.

6.1.1 Description of the acquisition system

The system uses a standard angiographic system that is equipped with an image intensifier on a C-arm which performs a rotational angiographic acquisition over a range of 180^o (see figure 6.3-middle), scanning 100 projection images. The clinician injects

Figure 6.3. 3D Rotational Angiography (3D-RA) System: Rotating around the patient, as shown in right image, the system acquire 100 projections along the half circular trajectory, as shown on the middle image; right image is a threshold-based volume rendering of a cerebral aneurysm computed from the 3D dataset reconstructed.

contrast product in the vessel during the acquisition (300 mg/ml iodinated contrast agent at a flow rate of 4-5 ml/s), to fill the pathology of interest during the scan. Acquired scans are then transfered to a workstation, and are automatically corrected for the image intensifier distortion, according to an initial performed calibration step. The reconstruction of the rotational images into a volume is performed by a modified version of the cone-beam algorithm of *Feldkamp* [51] (because of the half-circular trajectory). The default reconstruction procedure, from acquisition until the 3D volume creation takes approximately 6 minutes. And the 3D result can be viewed with a real-time volume rendering package (see figure 6.3-right).

6.1.2 Application to the segmentation of brain aneurysms

Threshold-based volume rendering is not sufficient to correctly extract valuable information from the dataset acquired (see figure 6.4 a cerebral aneurysm data, and its **MIP** projection image).

On complex objects, the topology will vary according to the selected visualization threshold on the raw volume. Some vessels will merge, and others split. It appears unreliable to visualize raw reconstructed volumes in terms of topology and vessel size.

We apply the competitive front method, described in section 5.3 to this kind of dataset, in order to operate a supervised fast pre-segmentation of the dataset. Only requirement is to set a seed region inside the object of interest, and another one in the background. The contrast-filled object being clearly bright on the dataset, it is not difficult at all to select a voxel v_{in} inside the aneurysm, and another voxel v_{out} outside it. Using our formulation of two propagating fronts $\mathcal{F}_{in}$ and $\mathcal{F}_{out}$ with two different initial penalty functions $\mathcal{P}^0_{in} = \mathcal{I}(v_{in})$ and $\mathcal{P}^0_{out} = \mathcal{I}(v_{out})$ lead to the segmentation shown in first row of figure 6.6. When the two fronts collides at time t,

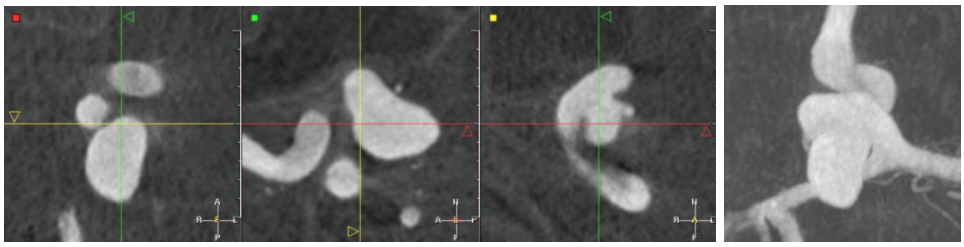


Figure 6.4. Example of 3D-RA dataset: The left image represents three orthogonal views of a cerebral aneurysm acquired with the **3D-RA** system; right image is a **MIP** of this dataset.

the new penalty $\mathcal{P}^t_{in}$ and $\mathcal{P}^t_{out}$ are input in our region-based geodesic active contour framework. Knowing the two distributions (μ_{in}, σ_{in}) and $(\mu_{out}, \sigma_{out})$, and that the aneurysms are brighter than the background, we initialize the region descriptors with the region following region probabilities:

$$\mathcal{P}_{in}(x,t) = \begin{cases} \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)}\ \exp(-\frac{(\mathcal{I}(x)-\mu_{in}(t))^2}{2\,\sigma^2_{in}(t)}) & \text{if } \mathcal{I}(x) < \mu_{in}(t) \\ \frac{1}{\sqrt{2\,\pi}\,\sigma_{in}(t)} & \text{if } \mathcal{I}(x) \geqslant \mu_{in}(t) \end{cases}$$

$$\mathcal{P}_{out}(x,t) = \begin{cases} \frac{1}{\sqrt{2\,\pi}\,\sigma_{out}(t)} & \text{if } \mathcal{I}(x) < \mu_{out}(t) \\ \frac{1}{\sqrt{2\,\pi}\,\sigma_{out}(t)}\ \exp(-\frac{(\mathcal{I}(x)-\mu_{out}(t))^2}{2\,\sigma^2_{out}(t)}) & \text{if } \mathcal{I}(x) \geqslant \mu_{out}(t) \end{cases}$$

The corresponding descriptors are shown in figure 6.5-right.

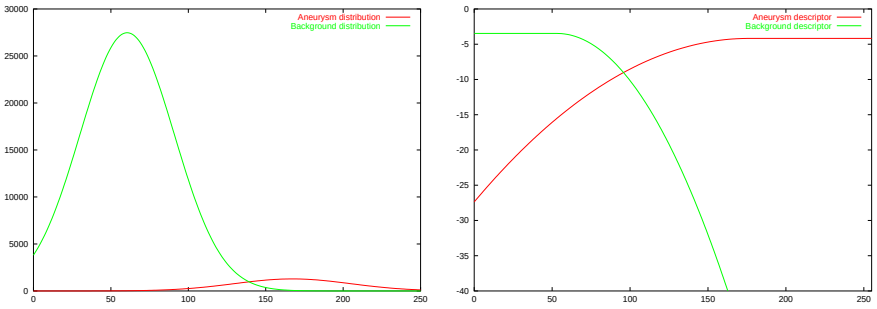


Figure 6.5. Region descriptors for the 3D-RA image: Left image shows the estimated Gaussian distribution of the inside and the outside of the cerebral aneurysm segmented by *Fast-Marching* as shown in first raw in figure 6.4; the inside $\rightsquigarrow \mathcal{N}(167, 37)$ and the background $\rightsquigarrow \mathcal{N}(60, 30)$.

Using the region descriptors k_{in} and k_{out} in the model defined in section 5.2, we iterate 20 times from the *Fast-Marching* initialization, to the final segmentation, shown in second row of figure 6.6.

The variability in positioning the initial seed voxels inside and outside the object does not have a clear impact on the final solution, since it provides an initial guess,

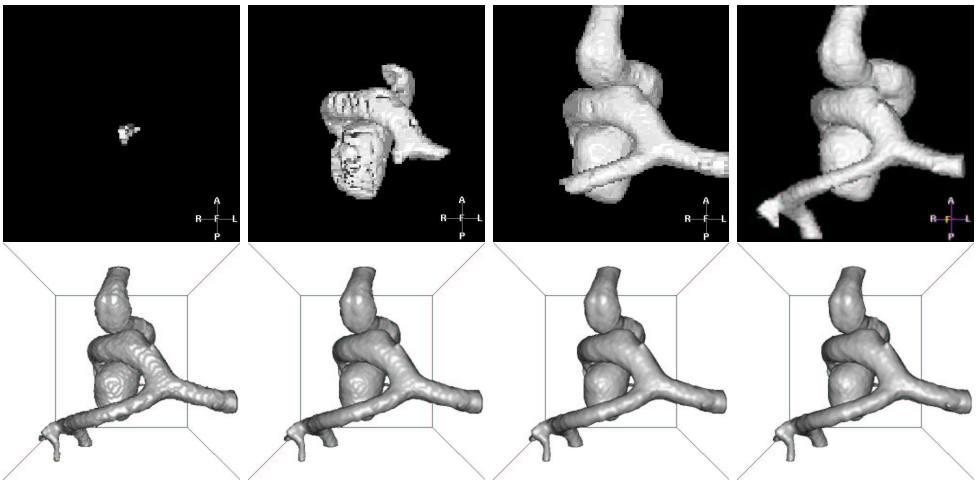


Figure 6.6. Example of brain vessels segmentation with combination of Fast-Marching and Level-Sets: First row shows iterations of the segmentation process described in section 5.3; second row shows iterations of the final refinement technique described in section 5.4; parameters for inside object $\alpha = \eta = .8$ and $\zeta = 0.05$.

near the final converged zero level-set, and since it is the initialization of the region-based descriptors. The independence of the region-based level-set framework from the initialization ensures that variability in setting the initial seeds.

The measure of the variability of the positioning of the seed point, whereas it does not have any clear impact on the final result, should be emphasized, since it is the initialization of the region-based descriptors. The variation in the acquisition protocol could optimized the settings of parameters of the boundary based forces, towards region-based forces importance.

And the setting of the seed points could be automated, since the dataset is centered in the volume of interest, and it always intersects the image borders (the intersection could be recognized, being the section of a circular vessel).

The same process, including supervised pre-segmentation and automatic final segmentation was applied to a set of cerebral aneurysms see figure 6.7. The descriptors k_{in} and k_{out} issued from the segmentation model of the cerebral aneurysm of figure 6.4 were used for each dataset. The boundary descriptors attracted the zero-level set of ϕ and converges rapidly. Parameterization was the same for each dataset, and stability of the protocol ensured fast convergence to the aneurysms shown in last row of figure 6.6.

The shape extraction of the aneurysm is done in less than 2 minutes on a 300MHz Sun Workstation. Knowing that the reconstruction procedure, from acquisition until the 3D volume creation takes 6 minutes, the added procedure value towards its cost is not important, especially if we consider that on a classical standard commercial PC, this cost could be divided by two. Therefore, this procedure provides all the

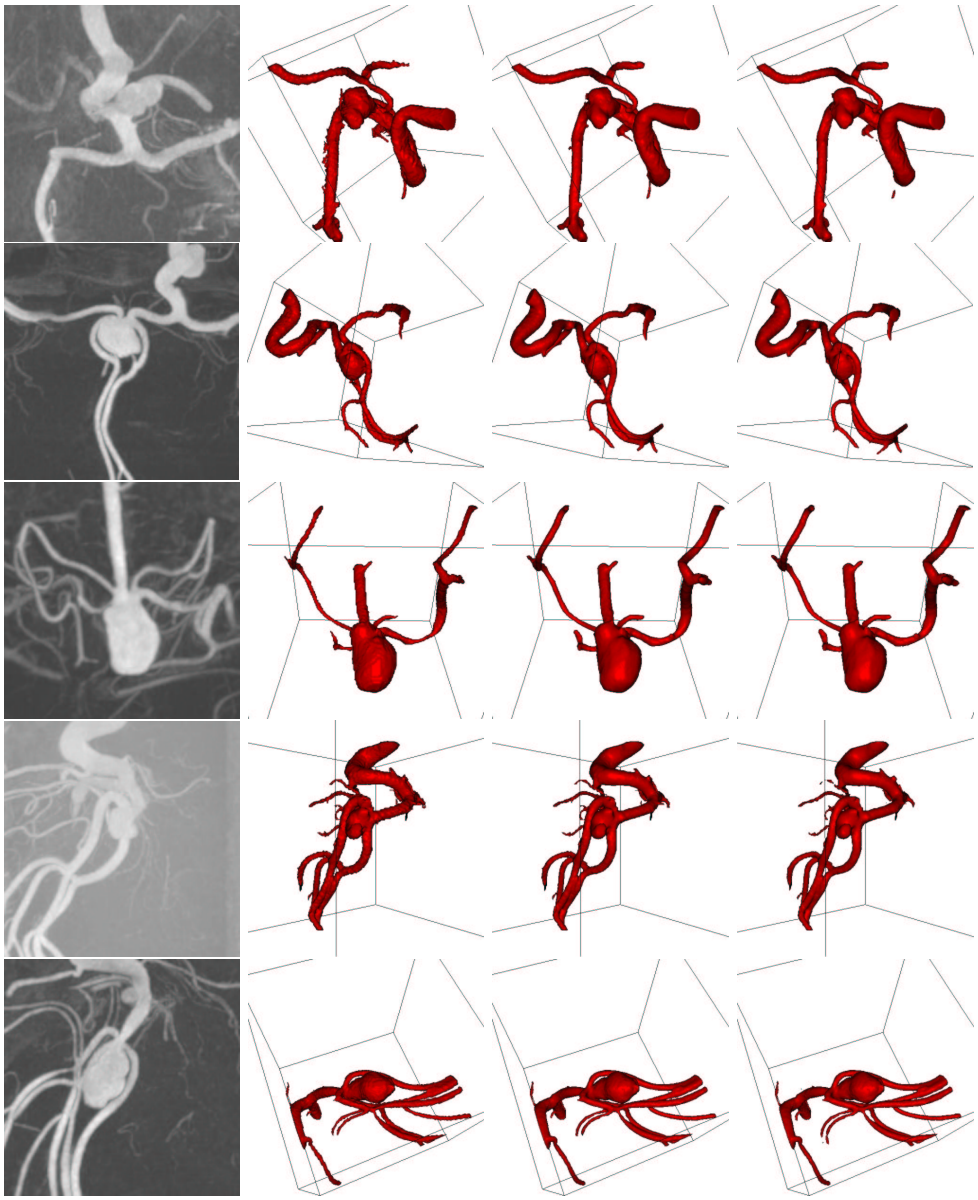


Figure 6.7. Segmentation of five 3D-RA datasets: First column contains the **MIP** images of the five data-sets; The surface rendered views show, from left to right in each row, iterations 0, 10 and 20 of the level-set segmentation process.

anatomical information required for the entire course of endovascular treatment of cerebral aneurysms. Direct benefits for interventional neuroradiology:

1. fast and accurate analysis of the morphology of the aneurysms, including determination of the size and the relationship with the parent vessel, allowing selection of the appropriate stent size, catheter and guide-wire thickness, or coil length;

2. fast and safe decision regarding the feasibility of endovascular approach.

3. to reduce radiation exposure: **DSA** images are still the gold standard in Angiography during interventional treatment, but when the 3D result is available, less 2D images need to be acquired before, during, and after the treatment;

6.1.3 Perspective

The 3D shape representation of the aneurysms creates lots of investigation fields. In particular we could reduce (or suppress) contrast agent injection: the 3D model could be mapped/projected on the 2D image during intervention, by registration techniques, and used as a mask for the treatment (see figure 6.8);

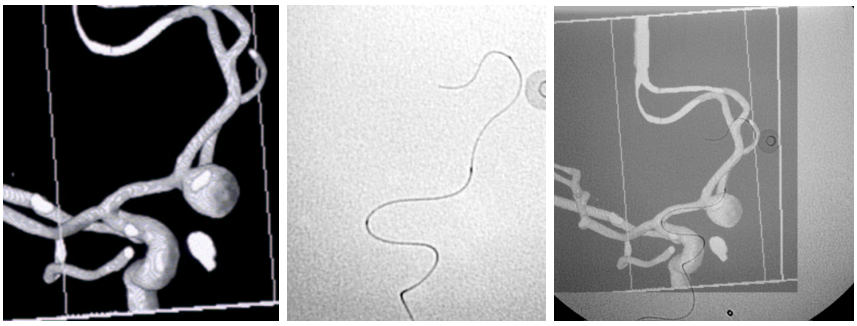


Figure 6.8. Live road mapping for interventional treatment: Left image is a 3D threshold based volume rendering; middle image is a 2D X-ray image of a guide and a catheter; right image is the superposition of those 2 images.

Another research direction could be the road-mapping of interventional treatment: the segmentation of the vessel structure enables to extract trajectories inside the vasculature, in order to efficiently reach the aneurysm (for stent placement for example), forecasting the future problems during intervention. Path extraction techniques, studied in chapter 2, could also be useful for extracting the optimal guide shape for an intervention: during intervention, for glue injection, the clinician introduces the catheter in the aneurysm, with the help of the guide. Then when the guide is extracted, the catheter - similar to a spaghetti - can move out the aneurysm, or eventually break it. This dangerous step can be avoided if the catheter has been optimally shaped using a trajectory artificially extracted in the 3D model.

Finally, the explicit surface extracted from the *Level-Sets* representation (with the *Marching-Cubes* algorithm) is a mesh that enables modeling of the blood flow inside the aneurysm. This model can help in preventing bursting of the aneurysms, but could be also very useful in following-up the result of an intervention.

6.2 Detection of Colon Polyps

This problem was already mentioned in section 3.1, concerning the path extraction tool developed specifically for virtual endoscopy. As said, colorectal cancer represents the third most frequently diagnosed cancer worldwide. If we consider malignant tumor, the yearly incidence of colorectal cancer probably approaches 160,000 cases [99]. This disease begins in the cells that line the colon, as polyps.

What is a Colon Polyp?

A polyp is a growth that occurs in the colon and other organs. These growths, or fleshy tumors, are shaped like a mushroom or a dome-like button, and occur on the inside lining of the colon.

What dangers do polyps present?

Colon polyps start out as benign tumors but in time may become malignant. The larger the polyp, the more likely it is to contain cancer cells.

Why do Colon Polyps and Cancer Form?

A great deal is known about why and how polyps form. There now is strong medical evidence that there are abnormal genes for colon polyps and cancer that can be passed from parent to child. Diet and foods may also be very important.

How are Colon Polyps diagnosed?

Importantly, colon cancer is one of the most curable forms of cancer. When detected early, more than 90 percent of patients can be cured. Early detection of colon polyps and cancer is performed usually with

1. study of the patient's medical history for identification of risk factors;
2. stool examination to detect occult blood from Colon cancers and large polyps;
3. visual examination of the lower colon using a lighted, flexible endoscope;
4. colonoscopy of the entire 5-6 foot long colon, under sedation;
5. x-ray exam (Barium Enema) which outlines the shadows of polyps and cancer;
6. virtual colonoscopy (already developed in section 3.1).

But still, even the *Virtual Endoscopy* relies on the user observation, for the detection, during visualization, of possible polyp existence. We already mentioned a possible unfolded view of the interior of the colon (see figure 3.21) that enables to see in all directions while traveling through the colon, but inspection remains a supervised process that rely on possible miss of hidden regions, from the camera point of view. Last drawback of endoscopy is that it relies on the choice of an opacity threshold input in the volume-rendering tool. The choice of this threshold critically constrain the position of the colon surface, thus the clinical validity of the observation.

6.2.1 Segmentation of the colon surface

We propose in the following to adapt the method developed in previous chapter, and already applied to cerebral aneurysm segmentation, to develop a initial framework of

semi-automatic polyp extraction. We further explore possibilities of detection with visualization techniques, using the curvature information of the object surface.

Classical CT scanner are generally very large. Instead of treating entire images, we used small volumes of interest which were selected by specialized physicians because of the presence of a particular pathology.

Before acquisition, the patient goes through a particular preparation during which the colon is emptied as much as possible. During the scan it is distended by inflating room air. The resulting image intensity in the colon lumen is rather uniform and lower than in the rest of the image, with a relatively good contrast. Therefore, the critical step of the segmentation process is the variability of the topology and geometry of the pathological structures.

Since the contrast is really important, as shown in figure 6.9-(a), it is a very easy task to set a seed point inside the colon, and another outside. Thus, supervised seg-

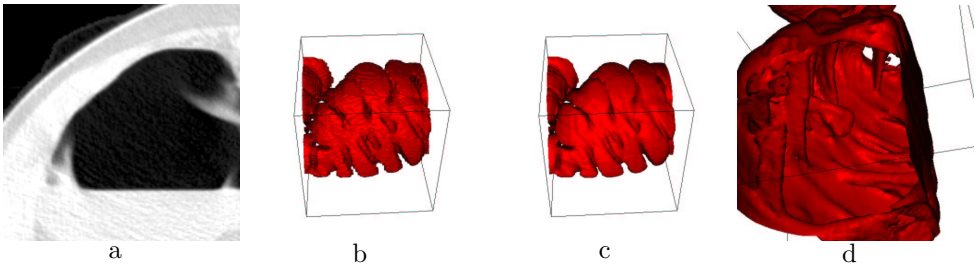
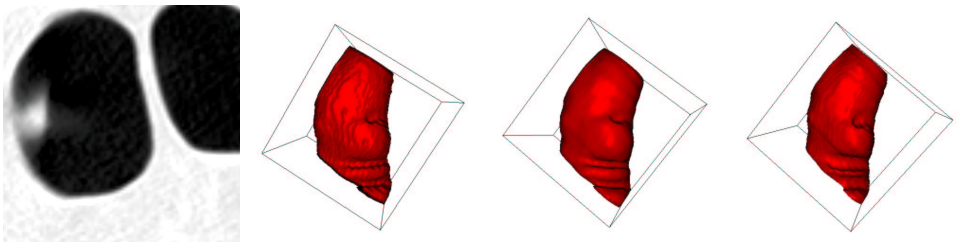


Figure 6.9. Example of polyps Segmentation: image (a) is a slice of a volume of interest (VOI) of the 3D CT scanner of the colon studied; image (b) shows the resulting pre-segmentation obtained with the *Fast-Marching* competitive version; image (c) is the result of the region-based *Level-Sets* at convergence, after 20 iterations; image (d) is an endoluminal view of the same segmented object, which emphasizes the polyps that grows on a fold of the colon surface.

mentation with front competition, using the *Fast-Marching* as detailed in section 5.3, can be easily achieved, as shown in figure 6.9-(b). Using the descriptors output by the pre-segmentation process, we initialize our *Level-Sets* with sigmoidal region-based forces. The justification of the use of those forces is the following: Pathological cases can arise, as shown in figure 6.10 In this application, the use of sigmoidal region-based forces is interesting because, due to the topology of the colon, that intersects several times the same volume of interest, it is possible to obtain disconnected parts of the colon in the same volume. Pre-segmentation being based on the setting of an interior seed point and an exterior seed point will lead to a binary image. This means that portions of the colon are probably included in the background. Therefore the statistical study of the background grey-levels will lead, with the Gaussian descriptors, to a background that have a large variance (see figure 6.11-left), whereas local histogram in the colon, due to the contrast, will give a small variance. Values for the example shown in figure 6.10-a, are $(\mu_{in} = 40, \sigma_{in} = 23)$ for the colon, and $(\mu_{out} = 620, \sigma_{out} = 385)$ for the background. Unless the user finds all disconnected parts of the colon, it is easier to use the *choice* of a darker region for the colon, and a brighter one, for the background, since evolution of the Gaussian model could lead to



a b c d

Figure 6.10. Segmentation tests with different descriptors: image a is the underlying dataset; image b is the initialization with the *Fast-Marching* algorithm; image c is the resulting segmentation after 100 iterations of the *Level-Sets* sigmoid region based forces; image d is the similar segmentation with Gaussian region based forces.

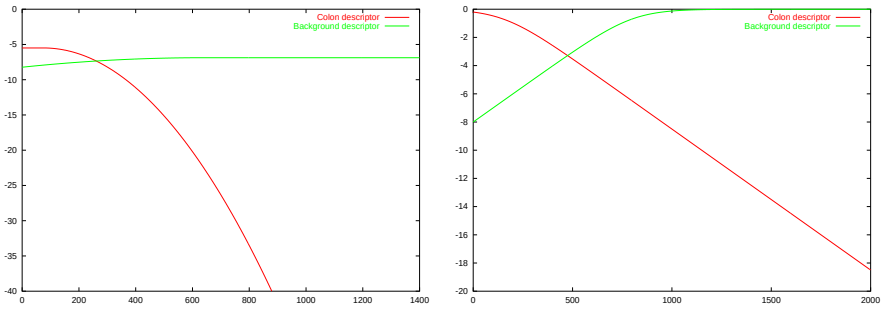


Figure 6.11. Region descriptors for the colon polyps: The left image represents the thresholded Gaussian descriptors for the colon polyps of figure 6.9; right image corresponds to the sigmoid descriptors of the same dataset.

unexpected results. In figure 6.10-4, the colon surface is flattened across iterations, because the variance of the background is higher. In figure 6.12, we display the result of the application of the same framework, using the same parameters than for example 6.9.

6.2.2 Visualization of the colon polyps

Colon polyps appear as convex regions in the lumen surface, in intraluminal 3D views (see segmentation results in figure 6.12). We tried to enhanced these suspect regions using a color information on the surface.

The specific shape of the colon polyps settle the use of the curvature information, mapped on the surface of the object, using an adequate color-map to highlight the cups. This technique has been already used in the surface of a segmented cortex, by *Zengh et al.* [195], using a measure defined originally by *Koenderink et al.* [94]. Using the values of $\tilde{\phi}$, $\phi(\cdot, t)$ at convergence, we know the expressions of the mean curvature κ_M and the Gaussian curvature κ_G for a surface propagating in three space

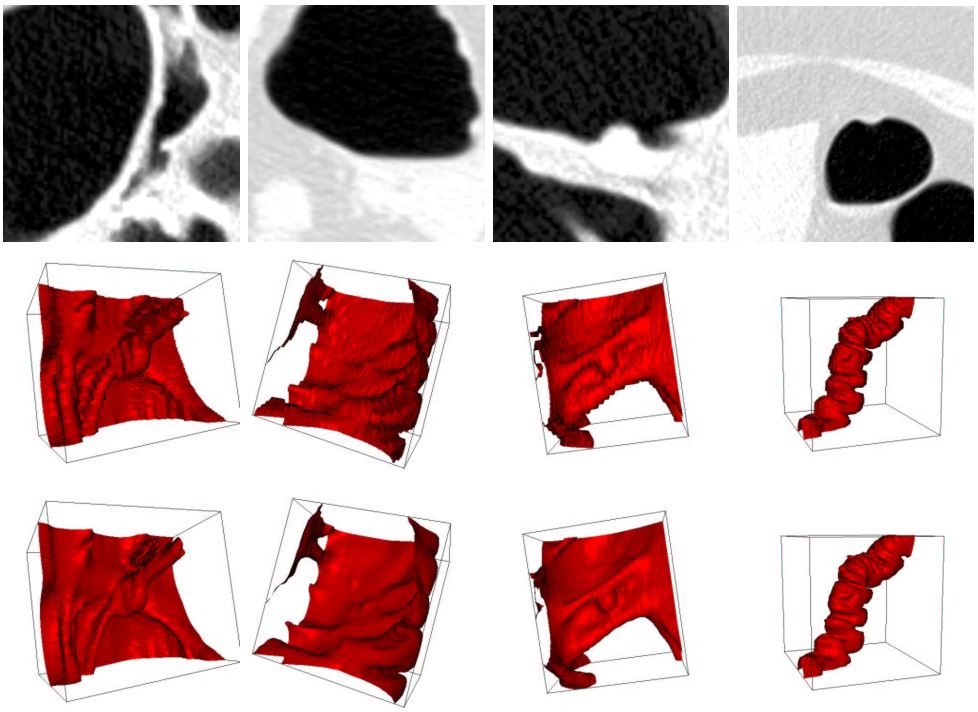


Figure 6.12. Polyps segmentation: First row: the four datasets used for segmentation; second row: the respective initializations given by the Fast-Marching competition algorithm described in section 5.3; third row: visualization after 20 iterations with region-based forces of the respective zero level-set.

dimension, in terms of the level-set function $\tilde{\phi}$. They can be easily computed using formulations given by *Sethian* [163]:

$$\kappa_M = \nabla \cdot \frac{\nabla \tilde{\phi}}{|\nabla \tilde{\phi}|} = \frac{\left\{ \begin{array}{c} (\tilde{\phi}_{yy} + \tilde{\phi}_{zz})\tilde{\phi}_x^2 + (\tilde{\phi}_{xx} + \tilde{\phi}_{zz})\tilde{\phi}_y^2 + (\tilde{\phi}_{yy} + \tilde{\phi}_{xx})\tilde{\phi}_z^2 \\ -2\tilde{\phi}_x\tilde{\phi}_y\tilde{\phi}_{xy} - 2\tilde{\phi}_x\tilde{\phi}_z\tilde{\phi}_{xz} - 2\tilde{\phi}_y\tilde{\phi}_z\tilde{\phi}_{yz} \end{array} \right\}}{(\tilde{\phi}_x^2 + \tilde{\phi}_y^2 + \tilde{\phi}_z^2)^{3/2}} \tag{6.1}$$

$$\kappa_G = \frac{\left\{ \begin{array}{c} \tilde{\phi}_x^2(\tilde{\phi}_{yy}\tilde{\phi}_{zz} - \tilde{\phi}_{yz}^2) + \tilde{\phi}_y^2(\tilde{\phi}_{xx}\tilde{\phi}_{zz} - \tilde{\phi}_{xz}^2) + \tilde{\phi}_z^2(\tilde{\phi}_{xx}\tilde{\phi}_{yy} - \tilde{\phi}_{xy}^2) \\ +2[\tilde{\phi}_x\tilde{\phi}_y(\tilde{\phi}_{xz}\tilde{\phi}_{yz} - \tilde{\phi}_{xy}\tilde{\phi}_{zz}) + \tilde{\phi}_y\tilde{\phi}_z(\tilde{\phi}_{xy}\tilde{\phi}_{xz} - \tilde{\phi}_{yz}\tilde{\phi}_{xx}) \\ +\tilde{\phi}_x\tilde{\phi}_z(\tilde{\phi}_{xy}\tilde{\phi}_{yz} - \tilde{\phi}_{xz}\tilde{\phi}_{yy})] \end{array} \right\}}{(\tilde{\phi}_x^2 + \tilde{\phi}_y^2 + \tilde{\phi}_z^2)^2} \tag{6.2}$$

We can use the scalars obtained and attach them to the vertices of the triangulated surface extracted by the *Marching-Cubes* (see figure 6.13-(b) for the surface extracted). However those values do not give valuable visible information that discriminates the structures we are looking for. We know that those convex structures

have the particularity to have high principle curvatures κ_1 and κ_2. Knowing that $\kappa_M = \kappa_1 + \kappa_2$ and $\kappa_G = \kappa_1 \times \kappa_2$, we deduce immediately the value of the principle curvatures.

We can map the maximum of κ_1 and κ_2 on the surface of the extracted object, as shown in figure 6.13-(c), but it does not give a clear view of the polyps.

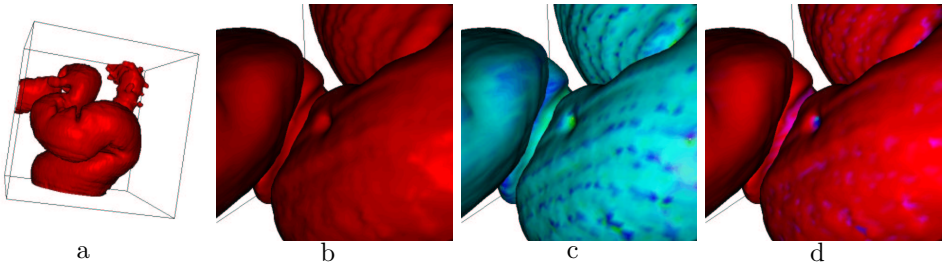


Figure 6.13. Example of polyps visualization: First image: the segmentation obtained by combining Fast-Marching and Level-Sets; second image: a zoom on a region of the colon volume; third image: the mapping of the function of the principle curvatures computed; fourth image: the threshold of this texture which highlights the polyps.

Furthermore, having in mind that we are looking for regions with negative curvatures, we apply the following equation

$$f(\kappa_1, \kappa_2) = \min(\max(\kappa_1, \kappa_2), 0) \tag{6.3}$$

which is interpolated at the vertices of the triangulated surface as shown in figure 6.13-(d). Only the regions where the two principle curvatures are negatives (*cups*) have negative f values, others get null values.

6.2.3 Perspectives

Unfortunately, other non-pathological regions are enhanced. The last row in figure 6.14 displays the result of this curvature mapping for four different datasets. Segmentation step was achieved using the same parameters for each datasets, and the curvature mapping is done with the same color-map. On several datasets, this mapping highlights other non-pathological regions: folds can be highlighted because of the sign of their principle curvatures. However our approach might be consider as a valuable start for the automatic detection of polyps, and currently viewed as an assisting tool for their visualization. The polyps are emphasized, and discriminated from the whole surface. Therefore, segmentation and visualization is achieved with a simple and fast process, leading to a pre-detection of the polyps which can already be used by any clinician.

In conclusion, the use of this kind of curvature filter outputs information relative to small and spherical polyps. Those polyps can grow and develop malignant tumor with non-smooth shapes where the curvature information is not suitable. Our tool finds its application in the early detection of the small polyps. The high precision of

the implicit level-set representation of the surface obtained through the segmentation process, enables to map on the surface informations for small objects like polyps. Our curvature measure is dedicated to this visualization.

Having in mind the settle of a non-supervised method of polyps detection, next step is recognition: Other non-pathological objects that are pre-detected can be discriminated with classification of the shapes of the connected components of the subset of the surface which corresponds to negative values of our function in equation (6.3). In last row of figure 6.14, we can imagine that we are able to unfold the surface of our colon, keeping the curvature information mapped on the explicit representation extracted at convergence of ϕ. This technique has been developed for level-sets techniques by *Hermosillo et al.* [76] using an explicit representation of the surface, with the curvature information mapped onto, which matches the implicit representation during its deformation (in this case, the mean curvature flow to unfold the brain surface, with a constraint on volume conservation). It was also developed in a different manner by *Bertalmio et al.* [12], where the region of interest is tracked as the intersection of two level-sets. Their application is to unfold the cortex in order to see the cerebral activity (mapped onto the surface with a given colormap) in the hidden sulci. Another possible technique is surface warping: using a warping based on registration methods , we can flatten the colon underlying triangulated surface of the zero level-set, into the plane, using a conformal mapping method, as in [73], or another mapping which preserve areas (as done in [74]).

Another possible development is the correct choice for the image scale. The image scale is an important parameter for the aspect of the surfaces and the regularity of the curvatures, but of course large smoothing factors can change the topology of the resulting segmented surfaces. The use of curvature flows seem unadapted as well [44]. A flow based on the intrinsic Laplacian of the mean curvature (see *Chopp and Sethian* [28]) might be an interesting tool to experiment in the future, but the numerical issues related to this flow are still to be explored, and no numerical scheme is currently available.

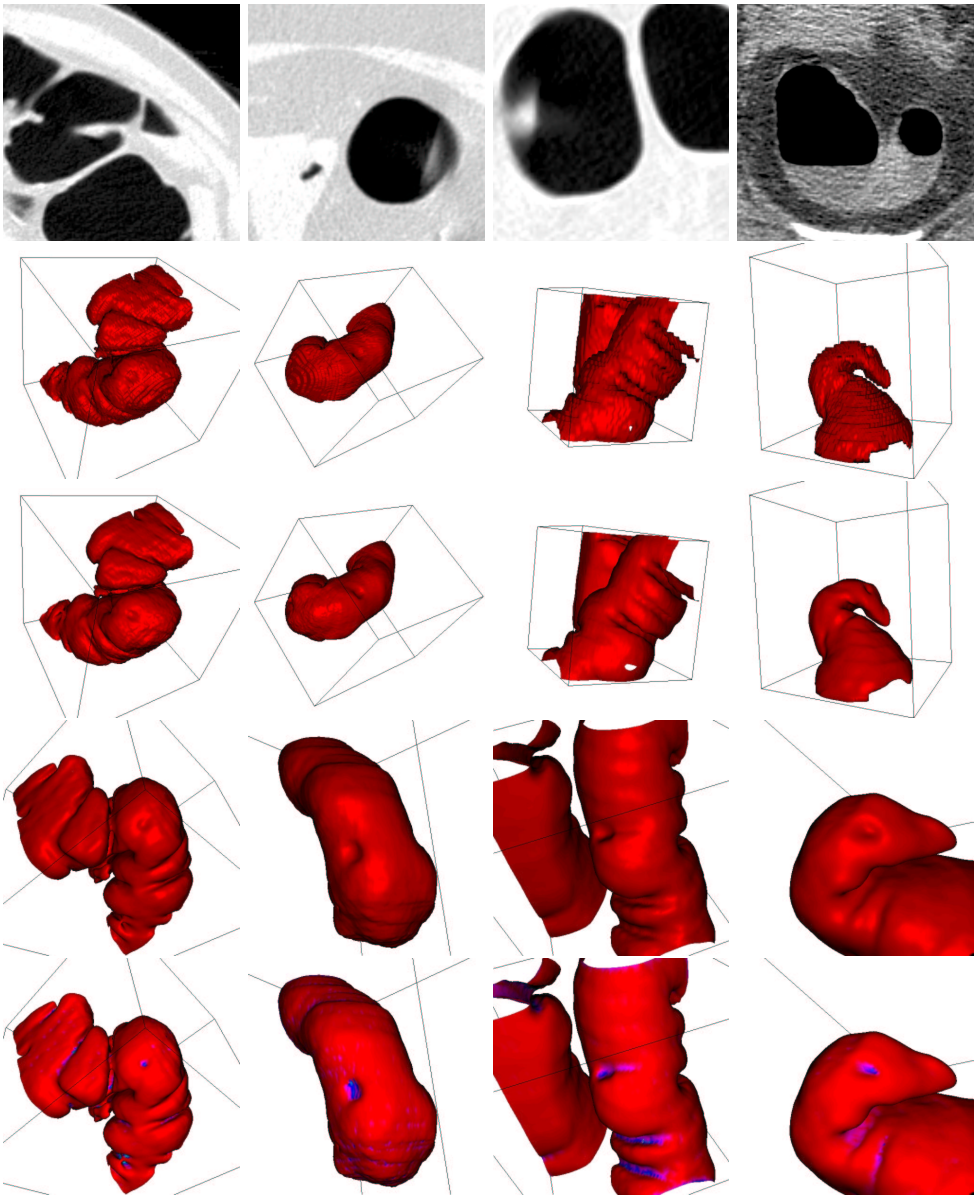


Figure 6.14. Polyps Visualization: First row: the four datasets used for segmentation and visualization; second row: the respective initializations given by the Fast-Marching competition algorithm described in section 5.3; third row: visualization after 20 iterations with region-based forces of the respective zero level-set; fourth row: another point of view for visualizing the polyps; fifth row: texture mapping with the curvature information.